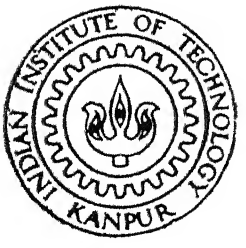
Gossiping in Multihop Radio Networks

by

NEERAJ GUPTA



DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

JANUARY, 1998

Gossiping in Multihop Radio Networks

A Thesis Submitted

in Partial Fulfillment of the Requirements

for the Degree of

Master of Technology

by

Neeraj Gupta

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

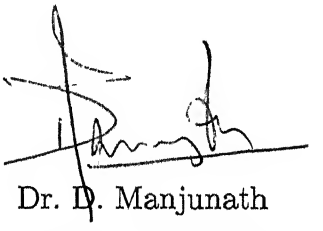
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

January 1998

Entered in System

Certificate

This is to certify that the work contained in the thesis entitled Gossiping in Multihop Radio Networks by Neeraj Gupta has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.



January 1998

Dr. D. Manjunath
Assistant Professor
Department of Electrical Engineering,
Indian Institute of Technology, Kanpur.

Acknowledgements

First of all, I want to express my sincere regards to the one who controls everything, for helping and supporting me in every moment of my stay at IITK.

Next, I would like to express my sincere gratitude to my thesis supervisor Dr. D. Manjunath for his invaluable guidance and constant encouragement throughout the work. I would like to thank him for giving me complete freedom in work. I am very thankful to him for giving me an opportunity to undertake this interesting topic as my M. Tech. thesis. It was his wonderful support, that enabled me to achieve the goal of this work.

My greatest thanks goes to Manishji and Amit who supported and helped me a lot in my difficult times. Special thanks to Qadeer, Farhan, Manoj and Vinay for their help and constant encouragement. I thank all my friends, especially those belonging to the Telecom group, for their memorable company during my stay at IITK.

No words can be said or written to express my feelings towards my family members. I am very thankful for their constant support and encouragement.

Last, but not the least, I would like to thank all the IITK community for providing me with a peaceful ambience and a home away from home.

Neeraj Gupta

Abstract

Gossiping is a specific information dissemination requirement in communication networks. In gossiping each node in the network has some information that it needs to communicate to everyone else in the network. This problem has many practical applications in communication networks where it is often necessary for each node to maintain up-to-date information regarding processor loads, lengths of queues, routing tables etc., of other nodes in the network.

Gossip algorithms have been studied for a variety of communication models like mail model, telephone model etc. However very little work has been done in understanding or developing gossip algorithms for radio networks.

In this thesis we first study gossiping in communication networks by conducting an extensive literature survey of the various communication models and the gossip algorithms therein. We then describe the problem of gossiping in radio networks using a graph theoretic framework and propose some good algorithms for gossiping in both the restricted and general topologies.

In the single frequency multihop radio networks that we consider, we assume that time is slotted and the system is synchronous. Moreover, in each time slot a station is permitted to transmit only one message out of the several ones that it may have at the time it is allowed to transmit in the network. For general topologies three algorithms for gossiping have been proposed each having its own advantages and disadvantages. We believe that the third algorithm that we propose, the Gather-Scatter algorithm, could really be among the best algorithms that can achieve gossip in radio networks. The specific topologies that we consider are the complete graph, star, ring and bus networks and these are chosen from the point of view of their applications in personal and indoor communication systems. We supplement the theoretical study of all our proposed algorithms with extensive experimental results. Our experimental results are based on an experimental model that is a realistic representation of real-life radio networks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Radio Networks

A radio network is a network of processors that communicate using radio signals. Typical examples include packet radio networks, cellular phone networks and satellite networks. If the network carries packets of data and uses the packet switching concepts developed for traditional (wireline) networks, then it is a packet radio network. Associated with a radio network is a set of carrier frequencies which are modulated to carry the packet data information. An important requirement for the modulation mechanism is to limit the bandwidth used by the signal of a specific carrier to within a prespecified band. In our work we will restrict ourselves to packet radio networks with one carrier frequency.

Packet radio networks are becoming popular as an obvious choice when the network needs to be deployed quickly. For example, it is widely believed that offering local access using a wireless medium will hasten the expansion of the telecommunication network in India. Packet radio networks are also an obvious choice in situations where the nodes are located in hostile and inaccessible terrain where the development of wireline networks is uneconomical. Networks with mobile nodes will of course need to have radio communication.

The characteristics of radio transmission and limitations of radio networks offer many challenging and interesting problems in their design. Firstly, the number of carrier frequencies available to a radio network is limited. Secondly, radio signals have a limited range over which they can be heard. Lastly radio networks operate in the broadcast mode

i.e., the transmission of a message issued by a network node reaches all nodes which are within "earshot" of the transmitting node. Multiple transmissions arriving concurrently at a given node will be interpreted as noise. This means that at any node in the radio network, interference from transmissions of other nodes can prevent it from concurrently receiving multiple transmissions. The medium access protocol for the network will take into account these limitations and allow the nodes to communicate among themselves.

In a radio communication network if all the nodes are within "earshot" of each other then it is a single hop network Since in this case all nodes can hear each other's transmission, there is no routing involved. The only issue that is of interest in the design of the medium access protocol for such a network is the set of rules governing the access to the medium. A variety of medium access protocols have been designed for such networks and reported in literature [1, 28, 34, 35]

Many a time, the radio network is such that the destination receiver is not within the range of the source transmitter. In such a situation, the message has to be received by intermediate hosts and transmitted to the destination through a store and forward operation. Such networks are called multihop radio networks. In this thesis our main interest will be on multihop networks. An interesting feature of multihop radio networks is that they allow "spatial reuse" of the channel, i.e., they allow, two network nodes that are sufficiently far apart in space to transmit simultaneously in time and not interfere with each others transmission. In this thesis we are primarily concerned with exploiting this spatial reuse feature of the network to increase the throughput of the medium access protocol.

Before proceeding further, let us first understand the interference phenomenon and the way it affects transmissions and receptions in multihop radio networks. The effects of interference can be explained through the *Hidden Node Problem* and the *Exposed Node Problem.*

Consider the example of the four node radio network shown in Fig. 1.1. A, B, C and D are the four nodes in the network. Nodes A and B are within each others' range and can interfere with each others' transmission if they transmit simultaneously. Node C can interfere with nodes B and D, but not with A. Let us assume that the nodes are using a Carrier Sense Multiple Access (CSMA) protocol i.e., they always listen

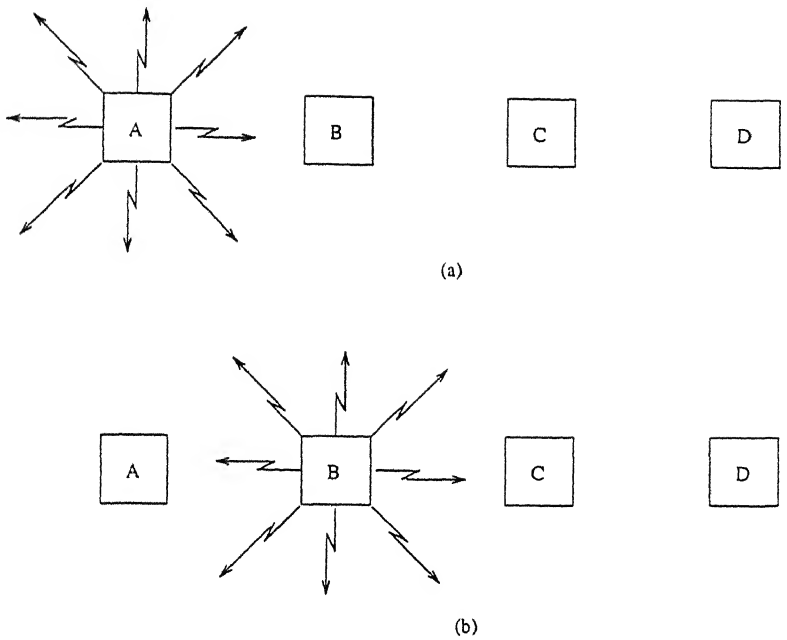


Figure 1.1: (a) Hidden Node Problem (b) Exposed Node Problem

on the channel and transmit when no other node is transmitting. Also assume that all communication requirements are point-to-point. Let us first consider what happens when A is transmitting a packet to B. This is shown in Figure 1.1(a). If C senses the medium, it will not hear A because A is out of its range, and thus it will falsely conclude that it can transmit. However, if C starts transmitting it will interfere with the transmission from A at B and B will not be able to receive the packet from either A or C. This will happen irrespective of the intended destination for C's transmission. This is the problem of a node not being able to detect the potential for interference from another node, a competitor for the medium, because it is too far away and is called the "Hidden Node Problem". Now let us consider the reverse situation: B transmitting to A, as shown in Fig. 1.1(b). If C senses the medium, it will hear an ongoing transmission and thus will falsely conclude that it may not transmit to D, when in fact such a transmission would cause bad reception only in the area covered by the ranges of nodes B and C where neither of intended receivers is located. This problem is more popularly known as the

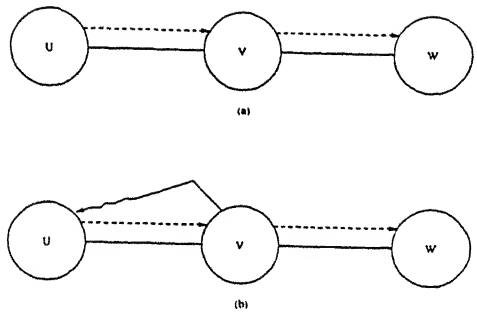


Figure 1.2: Collision occurring at v because its transmission time coincides with that of its one-hop neighbour u (a) Intended message transmission. (b) Actual signal propagation

"Exposed Node Problem".

Hence, for the transmissions in a multihop radio network to be intelligible to the intended receiver i.e., to guarantee that the message exchange is not affected by the above two problems, following two constraints must be satisfied.

- Transmission-Reception Constraint : A node can either transmit one packet or receive one packet at a time. Thus, it can neither transmit more than one packet simultaneously nor can transmit and receive simultaneously. If this constraint is violated, *Primary Interference* is said to have occurred.

- Interference Constraint: A node receiving a packet from a node A may not be within range of any other transmitting node B. If this constraint is violated, a collision occurs at the receiving node, nothing is received and *secondary interference* is said to have occurred.

These constraints can be better understood by going for an abstract representation of the radio network as a graph, $G = (V, E)$. In this representation, the N nodes in the network correspond to the vertices in the set $\{V\}$. Each directed edge $(u, v) \in E$ corresponds to the following relation between u and v: v receives every signal (message) transmitted by u. In other words, an edge $(u, v) \in E$ implies that v is in line of sight and within range of u. If the transmission and reception is symmetric, i.e., if $(v_1, v_2) \in E$ implies $(v_2, v_1) \in E$, then an undirected graph can be used to represent the radio network. From here on, we will use this graph representation to represent any sort of radio network.

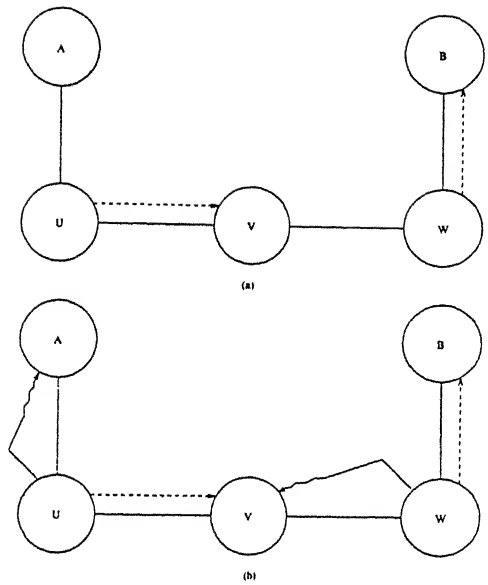


Figure 1.3: Collision occurring at v because the transmission time of two of its one-hop neighbours u and w coincide with each other. (a) Intended message transmission. (b) Actual signal propagation

Now, if we assume a time division multiplexed system in which one packet is transmitted in a slot, the above two constraints can be further summarised as follows: *The transmission time of a node should not coincide with its one-hop or two-hop neighbours.* Figures 1.2 and 1.3 illustrates these concepts more clearly. In the figure, the desired transmission of a message (signal propagation) is denoted by the dotted arrow, and the undesired signal propagation resulting from the broadcast property of the radio is denoted by the crooked arrow.

In Figure 1.2 as can be seen station u wants to transmit to station v and v wants to communicate with station w. However as can be seen if both u and v start transmitting simultaneously violating the condition that adjacent stations must be scheduled to transmit in different time slots, a collision occurs and nothing is received at v.

Similarly, in Figure 1.3, u wants to transmit to v and w wants to communicate with B. However if both u and w are allowed to transmit simultaneously, violating the condition of separating the transmissions of a node and its two hop neighbours in time, a collision occurs at station v. Moreover, station A also receives the packet from u irrespective of the fact that u's transmission was not meant for A.

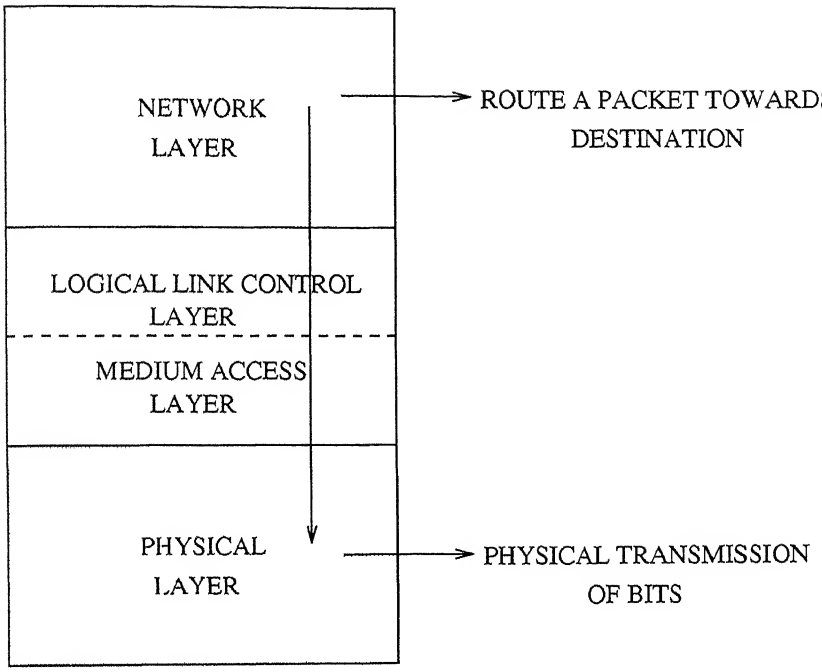


Figure 1.4: The lower layers of the ISO/OSI protocol stack. The services provided by the network and physical layers are indicated. The media access layer handles the scheduling of transmissions so that the network layer has transparent access to the medium.

As is quite obvious from the discussion in the preceding paragraphs, the key resource in radio networks is the broadcast channel which, at any time, may have multiple contenders (nodes) for its use. In such networks, also known as multiple access networks, the key issue is to determine who gets to use the resource, when it gets it and for how long it remains with it. All these issues are resolved at the Medium Access Control (MAC) layer of the protocol stack of the network and constitutes an important part in the design of a radio network. This is shown in Figure 1.4.

Traditional approaches to channel access may be categorized into random access and deterministic access techniques. In the random access approach, a node transmits a packet on the shared channel as soon as possible hoping that there is no interference from other nodes. If a collision occurs, a collision resolution mechanism is invoked to resolve it. In the deterministic access approach, each node is allocated a portion of the channel for its exclusive use to avoid collisions altogether.

Random access techniques are mainly used in networks having a large number of

nodes, each node having high peak and low average bandwidth requirements because in this case the interference probabilities will be low. Here, typically the total network throughput will also be much lower than the capacity of the physical channel. In contrast, deterministic access techniques are primarily used in heavily loaded networks and in applications where the channel access delay must be bounded. However the chief disadvantage of the deterministic access approach is that the channel allocated to a particular node cannot be used by other nodes even if that node has nothing to transmit, resulting in a wastage of channel bandwidth. However, for multihop networks, the deterministic access techniques have the potential to perform as well as the random access techniques. There are two reasons for this. Firstly, many random access techniques lose some of their efficiency because of the "Hidden Node Problem" explained previously. Secondly, the fixed access techniques gain efficiency since we now let nodes far apart share a channel (spatial reuse). Moreover, the deterministic access method provides channel control by explicit scheduling of transmissions, leading to better channel utilization and other associated benefits. Again, in a random access technique, the collision resolution scheme mainly determines the achievable throughput on the link, and it has been proved that any link throughput achievable by any collision resolution technique can also be achieved by time division scheduled access [8]. So, as a general rule, for networks with medium to heavy loads and bounded delay requirements, deterministic scheduling is really the preferred method of access [48]. In this thesis, our main interest will be on the design of deterministic schemes to allocate the radio channel in time and space among the different nodes in the network.

In the remainder of the thesis, we will assume that the partitioning of the channel is done according to time division multiplexing, time being divided into fixed length time slots. However, all the algorithms that we develop are also applicable to other deterministic channel partitioning methods like frequency division multiplexing.

Deterministic access using time division multiplexing requires the construction of a *basic schedule* (frame) which serves as a time table for controlling the activities of the nodes in the network. A frame consists of a number of equal duration time slots. Each time slot is equal to the message transmission time plus the maximum propagation delay between any two neighboring nodes in the network. In each slot, a node can be permit-

ted to either transmit, or receive or remain inactive in a manner that avoids collisions according to the needs of the particular communication application. By repeating the frame in time, a schedule for the entire duration of operation of the network can be obtained. Fortunately, in a multihop radio network, the spatial reuse property of the channel may be utilized, and more than one node can be scheduled to transmit in the same slot. This reduces the length of the frame required to schedule a given set of communicators. Shorter frames means a smaller inter-transmission delay for each node and hence results in an increased throughput. Thus the primary aim in the design of any deterministic access radio network is to schedule the simultaneous transmissions so as to obtain as short a frame as possible. In the next section we will discuss the various scheduling issues involved in any radio network design.

1.2 Scheduling in Radio Networks

As discussed in the previous section, scheduling of transmissions so that the network nodes can successfully exchange information in the presence of conflicts is a fundamental requirement in packet radio networks. Scheduling is basically a part of the medium access control (MAC) layer in the ISO-OSI/IEEE model and provides transparent channel access to the network layer. Depending on the nature of service required by the network layer, scheduling in radio networks can be of one of the two kinds - broadcast scheduling and link scheduling. In broadcast scheduling the transmission of a node is intended for and must be received collision free by all of its neighbors i.e., nodes within the receive-range. In link scheduling transmissions between a node pair are scheduled. The transmission of a node is intended for a particular node within receive-range and we require that there should be no collision at this receiver.

In addition to broadcast and link scheduling there can be special purpose schedules designed for specific communication patterns. The scheduling still involves the construction of a frame of fixed length time slots with a time table controlling the activity of each of the nodes in the slot. As before, the frame length is a measure of the "goodness of the schedule". In this case, depending on the nature of the communication pattern, certain kinds of primary or secondary conflicts may be permissible in the schedule. Examples of

such special purpose schedules are those constructed for applications like broadcasting (one to all communication), multicasting (one to many or many to many communication), gossiping (all to all communication), etc.

The scheduling algorithms, like any algorithm for a networked system, can be either centralised or distributed. A centralised algorithm uses all the information about the system for computation whereas in the distributed case the computation is distributed among the nodes in the network. The nodes typically use the local information available to them and participate in the computation by exchanging messages. Both centralised and distributed algorithms have been developed for radio network scheduling. Distributed scheduling algorithms, by distributing the task of scheduling among all the nodes in the network, are less prone to failure of a single node in the network. However, since the centralised solution uses a lot more information, it can produce better schedules than any distributed algorithm. In this thesis we will be mainly concerned with the construction of centralised scheduling algorithms for our applications.

Once an appropriate scheduling algorithm has been developed for a particular application we would obviously like to compare it with other known algorithms. For carrying out this comparison, the following two questions are of primary importance.

1. How good is the solution obtained by the algorithm?

2. How quickly is the solution obtained?

Let us consider these questions in detail.

The schedule length is the most relevant measure of the performance of a scheduling algorithm. In most networks the transmission schedule is constructed just once during initialization or during a topology change and the actual data communication is carried on with this schedule for as long as the network remains up or till the next topology change. This manifests the need for algorithms giving schedules with as small a frame length as possible.

The answer to the second question is given by the running time of the algorithm. The running time of an algorithm on a particular input is the number of primitive operations or "steps" executed. Every step is assumed to require a constant amount of time. The worst-case running time of an algorithm is the largest running time taken over all possible

inputs of a given "size" and bounds the time taken by the algorithm from above. In measuring the time complexity of an algorithm, rather than the running time, it is the rate of growth of the running time that is of concern in most cases. The upper bound on the rate of growth is expressed using the "big-oh notation". Thus if the running time of an algorithm, for an input of size n, increases as n^2, then the running time of the algorithm is $O(n^2)$. This notion captures the most significant factor in the rate of growth. We usually consider one algorithm to be more efficient than another if its running time has a lower order of growth, i.e., an $O(n)$ algorithm is better than an $O(n^2)$ algorithm. Algorithms can also be classified on the basis of their time complexities. A polynomial time algorithm has a running time of $O(p(n))$ where $p(n)$ is a polynomial in n. Any algorithm whose running time cannot be bounded by $O(p(n))$ is an exponential algorithm. Polynomial time algorithms are preferred in practical applications because of the property that polynomials are closed under composition. As is quite obvious these two requirements are not quite orthogonal and one may have to really look for algorithms giving a good tradeoff between the two.

1.3 Models for Information Exchange in a Network of Nodes

Let us denote by $\mathcal{N} = \{s_1, s_2, \cdots, \}$ the set of nodes in a network. Let a node $s_i \in \mathcal{N}$ have a message that it wants to send to some nodes in the set $\mathcal{R} \subset \mathcal{N}$. The two-tuple $SD_i = [s_i, \mathcal{R}]$ is called the SD_i-pair. At any time in the network, the information exchange requirement in the network can be defined by the set $IER = \{SD_1, SD_2, \cdots, \}$. In the above formulation, we have assumed that every source has exactly one message to transmit. This can be generalized to let source s_i have m, $m \in Z$, messages to send to $\mathcal{R}$. SD_i is now defined to be $SD_i = [s_i, m, \mathcal{R}]$ which means that s_i has m messages to send to the elements of $\mathcal{R}$. We can now define various information exchange models using the above notation. If the cardinality of $\mathcal{R}$ is always 1, then we have a set of *unicasts*. If $IER = \{[s_i, m, \mathcal{N}]\}$, for some i then we say that s_i has m messages to *broadcast* to the network. If $IER = \bigcup_{s_i}\{[s_i, m, \{r\}]\}\ \forall s_i \in \mathcal{N}$, then we say that node r is *accumulating* m messages from the network. If $IER = \bigcup_{s_i}\{[s_i, 1, \mathcal{N}]\}\ \forall s_i \in \mathcal{N}$ then the requirement

is of *gossiping*. Other information exchange problems can be similarly defined. In the following we will discuss the concepts of broadcasting, accumulation and gossiping in some detail.

As explained earlier, for broadcasting, one node in the network has some piece of information which has to be made available to all other nodes in the network. In accumulation all nodes except one has some piece of information which has to be made available to the one having nothing. Gossiping is a more complicated version of broadcasting in which each node in the network has a piece of information to be made available to all the remaining nodes. Broadcasting, accumulation and gossiping have numerous practical applications in computer networks where they can be used by each unit to maintain the latest status information about processor loads, queue lengths or routing tables at other nodes in the network. Other applications of these exchange models can be found in distributed databases and distributed computing. A comprehensive reference on broadcasting, accumulation and gossiping is available in [27].

For simplicity of presentation let us resort to a graph representation of networks as described below.

Any network can be represented by a graph $G = (V, E)$ where V is the set of nodes in the network and E is the set of edges in the graph. An edge is an ordered or an unordered pair, (n_i, n_j), where n_i and n_j are nodes in the graph G. An edge (n_i, n_j) is present in the graph representation of the network if node n_i can "directly communicate" with node n_j and vice versa.

We now explain the notion of what we call a *communication algorithm*. Any communication algorithm can be considered as a sequence of simple communication steps called *communication rounds*. The communication algorithm gives us complete information about the nodes which are going to communicate in a round and the information they are going to communicate. To specify the set of allowed communication algorithms a *communication model* is defined that specifies the way the communication links (edges in the graph representation), may or may not be used in a communication step. There are several communication models proposed and investigated in literature. Some of widely used ones are discussed below. We will use the graph notation in the discussion.

1. One-way Mode or Telegraph Communication Model

The edges in the graph represent a bidirectional link between nodes but communication between them is half duplex. Thus in a communication step, node x may send node y a letter containing a number of messages known to x or vice versa. Further, in this step, when x is sending a message to y or vice versa neither can be involved in any other activity (receiving from or sending to other nodes) in the network.

2. Two-way Mode or Telephone Communication Model

   The edges represent a bidirectional link between nodes and communication is full duplex, i.e., in a single communication step, nodes x and y may exchange all the information known to them in a manner equivalent to a telephone call between x and y. Again, neither x nor y can be involved in any other activity in the network in this particular step.

3. Full duplex, Delta Port Model or the Shouting Model

   In this model, the edges represent bidirectional links and a node can receive from and transmit to *all* its neighbours simultaneously in a communication step. A closely related model is the Half Duplex, Delta Port model where, as the name suggests, simultaneous transmission and reception are not allowed during a communication step.

4. Radio Network Model

   In this model it is assumed that nodes in the network communicate using radio transmitters. When a node transmits, all nodes within earshot will receive the message (barring situations where simultaneous transmissions lead to primary and secondary interference). We will be primarily concerned with this model in this thesis

Many other models have also been defined and proposed but most of them can be considered as generalizations of the above ones. These models are discussed in detail in Chapter 2.

A further specification of the activity in a communication step is the amount of information that can be exchanged between the communicating nodes which can be either bounded or unbounded. In the telegraph and the telephone model it is reasonable

not to bound the amount of information exchange. However, in the shouting and the radio network model, that would really be an unreasonable assumption. Most of the previous work on gossiping in radio networks assumes that the amount of information that can be exchanged during a communication step is of an arbitrary unbounded length. This means that under this assumption the messages available at a node at any time during the execution of a protocol can be freely concatenated and the resulting packet can be transmitted in a constant amount of time !. That is, it has been assumed that the time required to transmit a message is independent of its length !. This is clearly an unreasonable assumption and in this thesis we relax this assumption to provide algorithms for gossiping in a more realistic situation in which exactly one message is exchanged in a communication step.

1.4 Overview of the Thesis

In this thesis we consider the problem of gossiping in multihop radio networks for the case when the communicating nodes can exactly exchange one message in a transmission slot. It is assumed that the system is a time slotted synchronous system.

The problem of gossiping has been studied in the past for a variety of system models like telegraph model, telephone model, shouting model etc. However in most of the models the problem has been considered from the point of view of its application in parallel processing, in which restricted classes of graphs are more popular in practice. Thus some very good algorithms for gossiping in restricted classes of networks for these models have been developed. In contrast to these models, however in the case of multihop networks the one essential element which complicates matters is that, contrary to point to point networks in which each channel is utilized by a single pair of nodes, the radio channel in packet radio networks is a "multi-access broadcast" resource. That is in a given locality determined by radio connectivity:

- The channel is shared by many contending users thus demanding the need for some good scheduling techniques
- Radio is a broadcast medium and thus the action taken by a node has an effect on the actions taken by neighbouring nodes and their outcome

Taking all the peculiar properties of radio networks into account, in this thesis, we propose some good algorithms for gossiping in networks with some regular topologies as well as for arbitrarily connected radio networks. The choice of these regular topologies is made by keeping in view their importance in personal and indoor communication systems.

The remainder of the thesis is organized as follows. We begin in Chapter 2 by giving a brief history of the gossip problem and then take a look at some of the work done to solve this problem under various communication models. In chapter 3 we address the problem of scheduling in radio networks, giving the need for scheduling and various existing scheduling techniques in practice. Chapter 4 is devoted to the description and performance analysis of algorithms for gossiping in regular and arbitrary topology networks. Chapter 5 concludes the thesis and gives directions for future extensions of our work.

Chapter 2

Information Exchange Models and Gossip Algorithms

2.1 Introduction

As was mentioned in Chapter 1, three interesting information exchange requirements in communication networks are broadcast, accumulation and gossiping. In broadcasting, a node in the network has some information which it has to distribute to all other nodes in the network. In accumulation, all nodes have some information and a designated node (sink) has to collect the information from all the sources. In gossiping (also called total-exchange or all-to-all communication) there is some information at every node that has to be diffused to all the nodes in the network. In this thesis our interest is in gossiping which abstracts a variety of communication problems in a large class of communication systems and networks.

Gossiping finds applications in a large number of vastly different areas. For example, gossiping is a fundamental primitive in distributed memory multiprocessor systems where it can be used, for example, for global processor synchronization. Also, it is an integral part of a large class of parallel computation problems like linear system solving, Discrete Fourier transform and parallel sorting where both input and output data are required to be distributed across the network. Gossiping can also be used to study load balancing, congestion, control, routing control, synchronization and billing in networked communication. Updating of distributed databases can also use gossip algorithms. Thus,

we see that gossiping is very rich in practical applications. It has been widely studied under a variety of communication models some of which have been briefly described in Chapter 1.

We will first briefly discuss the development of the gossip problem. It all started in 1950, when Bavelas [5] studied the effectiveness of different communication patterns in helping small groups of people solve common tasks. One of the tasks that was studied was the following : five players are dealt five playing cards each; the cards may not be passed around, but the players are allowed to communicate with one another according to a given set of rules by writing messages. The task is considered finished when each player selects one of his five cards so that the five cards together forms the best poker hand that can be made by selecting one card from each person. To measure the efficiency of the algorithm, the number of messages and the time required to complete the task were used as metrics. Bavelas showed that in a group of n players if any communication pattern is allowed and each message takes unit time, then the time required to complete the task is no more than $\log_2 n$ time units.

In 1951, Shimbel [49] proposed the following group communication problem as an application of matrix algebra. There are n members in a group. Each member knows initially to whom they can send messages but does not know to whom other members can send messages. The problem is for each member to learn "the complete structure of their communication system". Shimbel had several interesting posers in this problem: find the structures that allow solution of the problem; find the minimum time required to solve the problem; the tradeoff between the number of channels and the time required. In 1954, Landau [39] studied task-oriented groups in which each individual in a group, initially has one piece of information which must be transmitted to all the others in order to complete a task. At every sending time, the individuals send all the information they have acquired to one other individual. A typical example of a task in this case is the following: "Each individual is given a set of coloured marbles, only one colour being common to all the group. The members of the group must exchange messages about their own colors and what they have learned about the colors of the others until everybody knows the common colour. Messages are sent only after everyone has indicated readiness to transmit: the transmissions then take place simultaneously, each individual sending to

just one of their possible recipients. Each member knows initially to whom they can send messages, but does not know to whom the others can send. A major assumption that Landau made was that each individual picks a recipient at random from among those to whom they are permitted to send messages, with equal probability of picking any one of them. For a variety of communication patterns among three and four people, Landau determined the expected value of the time it would take to complete such a task. A natural progression in the study of the gossip problem occurred with Hajnel, Milner and Szemeredi [32] attributing the following problem to Boyd: There are n ladies, and each one of them knows an item of scandal which is not known to any of the others. They communicate by telephone, and whenever two ladies make a call, they pass on to each other, as much scandal as they know at the time. The number of calls needed before all ladies know all of the scandals is to be determined. This problem, which has become known as the *Gossip Problem*, or the *Telephone Problem*, has been the inspiration for many research papers that study problems concerning the spread of information through telephone calls, conference calls, letters or even computer networks.

To solve the gossip problem for a given communicating group (modelled by a connected, undirected graph G) a communication strategy such that all vertices in G learn the cumulative message of G must be found. A communication strategy is realized by a gossip algorithm consisting of a number of communication steps that we call *rounds*. The rules describing what can happen in one communication step or a *round* are defined by the communication model. During a communication step, we can either bound the amount of information sent from the sender to the receiver or leave it unbounded which essentially assumes that any amount of information is exchanged in one unit of time.

Cases where the assumptions of unbounded information exchange in a communication step would be valid include the following: systems where the messages are small and the dependence of transmission time on message size is negligible; systems in which it is possible to coalesce the available information (for example, when a global sum or maximum is being computed); systems where the transmission hardware or software uses very large packet sizes; systems where local per-message overhead is high enough that actual message transmission time is negligible. The assumptions of an unbounded information exchange in a communication step is clearly unrealistic in, for example,

computer networks where the time required to gossip with fixed size packets is a truer measure and gives a good estimate of the real time needed to disseminate information in the network.

Now the question of measuring the efficiency of a gossip algorithm arises. To answer this it is first necessary to understand the concept of a round more clearly. In the context of gossiping, a round consists of a set of transmissions that can be performed simultaneously. These are entirely decided by the communication algorithm on the basis of the communication model being used. The number of rounds consumed in completing the gossip is a measure of the time that would take to achieve gossip. Therefore, one obvious measure to evaluate the efficiency of a communication algorithm is the number of rounds required to complete the gossip. A second measure is to determine how long it would take to decide the transmitters and the messages that they are to transmit in each round. We will use both these measures in studying the efficiency of our gossip algorithms.

After having explained the necessary preliminaries let us now turn our attention to the various communication models and gossip algorithms in these models. Unfortunately in most of the models the problem of "gossiping in minimal time" has been proved to be NP-hard. This suggests two directions in which one might profitably proceed. One is the development of efficient approximation algorithms for gossiping in communication networks modelled by arbitrary undirected graphs. The second direction would be to consider the problem of gossiping in communication systems that can be modelled by specific classes of graph architectures and then find the algorithm to achieve gossiping in minimal time in these architectures. Again, the direction to be chosen is dictated by the particular application. For example, in certain applications like parallel and distributed computing the second approach could be more useful because the goal here is to investigate the advantages and disadvantages of a particular interconnection structure with respect to information dissemination based on which the suitability of this network for parallel computing is decided. However, for general wireline/wireless networks the first approach would definitely be more useful as the main interest here is to find out how quickly the information can be updated among the network nodes using a particular communication model, whatever distribution of nodes the network may have. The

communication algorithm here should be able to effectively perform the task irrespective of the network configuration.

As explained in Chapter 1, there are four widely used communication models for information dissemination namely mail model, telephone model, delta port model and radio network model. Our approach in the next few sections will be to first explain these models in detail and then give a brief overview of the best known gossip algorithms for these models for both the bounded length and the unbounded packet length approach.

2.2 Mail Model or Telegraph Communication Model

One of the most widely studied communication models is the Mail Model. Because of its half duplex nature (one-way mode) it is also known as the H_1 model. The subscript 1 indicates that a node can only communicate with one other node at any given time.

In a distributed memory multiprocessor, a communication transaction is an interplay of both hardware and software activity. The H_1 model of communication makes the weakest assumptions about both the hardware and software capabilities. Actually this model represents the minimal communication capability that a loosely coupled multiprocessor could possibly have. The upper bounds developed under the H_1 model apply to all models, while lower bounds under H_1 serve as upper bounds for potential lower bounds under other models. It characterizes one real parallel processing machine fairly well - the first generation NCUBE, especially for small messages. Algorithms developed under the H_1 model will generally impose the least load on a system since they represent the smallest use of communication resources.

In the mail model, in a single round, each node may be active through one of its adjacent edges either as sender or as receiver, i.e., corresponding to the half duplex mode of operation, the information flow is only one-way. Thus, a one-way communication algorithm for a graph G=(V, E) can be described as a sequence $E_1, E_2, \ldots, E_m$ of sets (matchings) $E_i \subseteq \vec{E}$, where $\vec{E} = \{(v \rightarrow u), (u \rightarrow v) \mid (u, v) \in E\}$ and if $(x_1 \rightarrow y_1), (x_2 \rightarrow y_2) \in E_i$ and $(x_1, y_1) \neq (x_2, y_2)$ for some $i \in \{1, 2, \ldots, m\}$, then $x_1 \neq x_2 \wedge x_1 \neq y_2 \wedge y_1 \neq x_2 \wedge y_1 \neq y_2$.

To explain it more clearly an accumulation algorithm for a graph of 7 nodes arranged

in a straight line as shown in Figure 2.1 is described below. The accumulation is at node x_4.

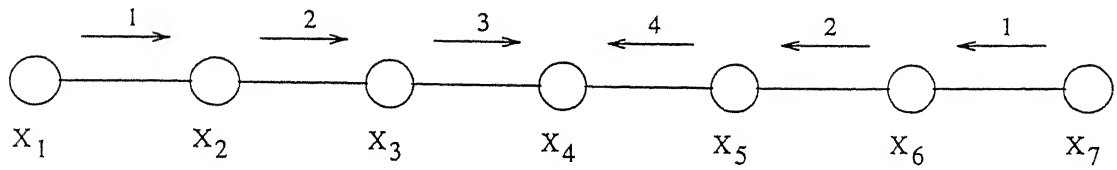


Figure 2.1: Accumulation algorithm for a Bus network of 7 nodes in the Telegraph model

In the first round the node x_1 sends all the information that it has to x_2, and x_7 sends all of its information to x_6. In the second round x_2 sends to x_3 and x_6 sends to x_5. In the third round x_3 sends to x_4, and finally in the 4th round x_5 sends all of the information it has to x_4 (assuming transmissions of unbounded length). Obviously this communication algorithm can be described as follows:

$\{(x_1 \rightarrow x_2), (x_7 \rightarrow x_6)\}, \{(x_2 \rightarrow x_3), (x_6 \rightarrow x_5)\}, \{(x_3 \rightarrow x_4)\}, \{(x_5 \rightarrow x_4)\},$

We can easily see that all the properties of the one-way mode are satisfied.

Having explained the model in detail, we now give a brief description of the various gossip algorithms in this model. In almost all of the proposed algorithms it is assumed that the scheduling of transmissions during each round is centrally controlled by a supervisor who possesses complete knowledge of the network.

2.2.1 Unbounded Information Exchange in a Communication Step

Unfortunately the first result available here is :

> The problem of determining an optimal scheme (in terms of number of rounds needed) for gossiping in a general graph is NP-hard [3].

However, because of a variety of applications of this model in parallel computing problems, nearly optimal algorithms have been given for specific architectures like complete graphs, hypercubes, buses, grids, rings and trees. Let us consider them one by one. Let us denote by $r_{H_1}(G)$ the necessary and sufficient number of rounds for gossiping in this model.

The most important previous work on gossiping in a mail model has been for the complete graph K_n. Analysis of the gossip problem for the complete graph is important because it provides a benchmark for all the other graphs. It is quite easy (using a spanning binomial tree in K_n) to complete gossiping in K_n in $2\lceil \log_2 n \rceil$ rounds. In 1979 Entringer and Slater [21] gave an algorithm which accomplished gossiping in K_n in just $\log_\phi n + 5$ rounds, where $\phi = \frac{1}{2}(1+\sqrt{5})$. Even and Monien [19] used an elegant matrix norm argument to show that any algorithm for gossiping in K_n requires at least $\log_\phi n + 2$ rounds, thus establishing that the Entringer-Slater algorithm is asymptotically optimal.

For a hypercube H_k there are several one-way gossip algorithms working in $2k$ rounds (see for instance [16]), and some researchers have also conjectured that $r_{H_1}(H_n) = 2n$ [3]. Surprisingly, Krumme [37], has found a one-way gossip algorithm for H_9 working in 17 rounds. The generalizations of this algorithm for large hypercubes has led to a gossip algorithm working in $1.88k$ rounds [37]. However, the highest lower bound on $r_{H_1}(H_k)$ is known to be $1.44k$. This gap between $1.44k$ and $1.88k$ leaves enough space for further investigation Again, no conjecture has been known so far concerning the placement of $r_{H_1}(H_k)$ between $1.44k$ and $1.88k$, i.e., it is not known whether there is a greater chance to improve the lower bound than to improve the upper bound or vice versa.

In the case of multidimensional grids the relatively large diameter makes it possible to perform one-way gossip optimally in the number of rounds equal to the diameter. This result is established for $n_1 \times n_2 \times \cdots \times n_k$ grids for any $k \geq 2$ and for any $n_i \geq 9$ [15]. In [3] it is shown that for $n \times m$ grids, where $n \geq 6$, if d is the diameter of the grid, one-way gossip is possible in $d+1$ rounds for even values of m and $d+2$ rounds for odd values of m. For several small grids the problem to find the optimal gossip algorithm is still left open.

For a bus B_n consisting of n nodes, following results are known.

- $r_{H_1}(B_n) = n$ for any even integer $n \geq 2$ and
- $r_{H_1}(B_n) = n+1$ for any odd integer $n \geq 3$

Optimal algorithms for these cases can be found in [27].

Let us now consider rings. Let R_n denote a ring with n vertices. Some of the best known bounds for the problem were $\frac{n}{2} \leq r_{H_1}(R_n) \leq \frac{n}{2} + O(\sqrt{n})$ [3]. Juraj, Claus and Monien [33] improved these bounds and gave the following results

- $r_{H_1}(R_n) = \frac{n}{2} + \lceil\sqrt{2n}\rceil - 1$ for each even $n > 3$, and
- $\lceil\frac{n}{2}\rceil + \lceil\sqrt{2n} - \frac{1}{2}\rceil - 1 \leq r_{H_1}(R_n) \leq \lceil\frac{n}{2}\rceil + \lceil 2\sqrt{\lceil\frac{n}{2}\rceil}\rceil - 1$ for each odd $n \geq 3$.

Nearly optimal algorithms for rings can be found in [33].

For a tree T of n vertices, Labahn [L80] showed that $2\lceil\log_2 n\rceil \leq r_{H_1}(T) \leq 2(n-1)$, and characterized trees which allow gossiping in $2\lceil\log_2 n\rceil$ rounds as the subtrees of the binomial tree. In 1990, Hakimi et al [3] determined $r_{H_1}(T)$ precisely, and showed that an accumulation/broadcast scheme will be optimal for trees.

Algorithms have also been proposed for some other special classes of graphs like Cube Connected Cycles, Butterfly Networks, Shuffle-Exchange Networks and DeBruijn Networks. For more details on these see [23, 33], and related papers.

Although the problem of finding an optimal gossiping algorithm for arbitrary connected graphs has been proved to be NP-hard, the following approximation algorithm has been found to be quite efficient. Choose a spanning tree of height R, send all messages to the root and then reverse the process to distribute the collection to all nodes. To construct the spanning tree, choose the centre of the graph as the root. Then in a breadth-first manner add nodes at distance $1, 2, \ldots, R$, adding just one edge when adding a node so that the constructed graph is always a tree.

2.2.2 Bounded Information Exchange in a Communication Step

It is known that determining the optimal number of rounds for gossiping with unbounded information exchange in general graphs is NP-hard; almost certainly the problem remains NP-hard when the amount of information exchanged in a communication step is bounded, although this has not been proved and remains an open problem. Again, because of several applications of this model in various areas like parallel computing some gossip algorithms have been proposed in literature.

Let $r_{H_1}(p, G)$ denote the number of rounds needed to gossip in a graph $G = (V, E)$ using packets of size p, implying that atmost a total of p messages can be transmitted by a node in each communication step. Before considering the various gossip algorithms, the following easy lower bound on $r_{H_1}(p, G)$ that is nonetheless quite useful, can be obtained.

$$r_{H_1}(p, G) \geq \frac{|V|(|V|-1)}{p\lfloor\frac{|V|}{2}\rfloor} \tag{2.1}$$

and the inequality is strict for $p > 1$.

To give an idea of the proof of this inequality, it can be seen that each vertex must receive $|V| - 1$ messages and thus a total of $|V|(|V| - 1)$ messages must be received by the vertices. In a single round, a total of at most $p\lfloor\frac{|V|}{2}\rfloor$ messages can be received. Thus it can be seen that $r_{H_1}(p, G) \geq |V|(|V| - 1)/p\lfloor\frac{|V|}{2}\rfloor$. The inequality is strict for $p > 1$, since no vertex can send p messages in the first round.

Let us now consider how to gossip with bounded information exchange in various regular graphs.

In complete graphs, for the case $p = 1$, Bagchi, Schemeichel and Hakimi [4] gave an algorithm to gossip. The algorithm proposed completes gossiping $2(n - 1)$ rounds for even values of n and $2n$ rounds for odd values of n. They used the following simple parallel algorithm:

```
begin
    Edge color K_n with X'(K_n) colors.
    for j=1 to X'(K_n) do
        for each edge(u, v) do in parallel
            parbegin
                u sends his initial message to v during round 2j − 1
                v sends his initial message to u during round 2j
            parend
        endfor
    endfor
end
```

For the case of $p > 1$, another algorithm has also been proposed that is optimal to within a small constant. Relevant bound given is that for $p \geq 2$,

$$r_{H_1}(p, K_n) > 2n/p + 1.44(\log_2 p) + O(1)$$

Let H_d denote the d-dimensional hypercube with $n = 2^d$ vertices. An efficient algorithm for gossiping in this structure has been proposed in [4], which is close to optimal, especially when p is a power of 2. The following results have been established

$$r_{H_1}(p, H_d) \leq \frac{2n}{p} + 2(\log_2 n) - \frac{2}{p} \qquad \forall p$$

$$r_{H_1}(p, H_d) \leq \frac{2n}{p} + 2(\log_2 p) - 2 \qquad \text{for } p = 2^k, k = 1, 2, \ldots$$

For a bus B_n with n vertices, it has been shown in [4], that for $p = 4$, one can gossip in B_n in $\text{diam}(B_n) + O(1)$ rounds. This is optimal upto an additive constant since $r_{H_1}(p, G) \geq \text{diam}(G)$ for any p and G. For $p = 3$ a slightly different approach from above yielded $r_{H_1}(3, B_n) \leq \frac{7}{6}n + O(1)$

For $p = 1$ and 2, the strategy used to gossip can be summarized as follows:

Assume that n is even. Accumulate at the $(\frac{n}{2} + 1)$ vertex from the left end (denote by v_0) and then broadcast from this vertex information from each half of the bus to the other half. At the same time, the information of the vertices in each half are to be disseminated to the other vertices in the same half. Following this simple procedure it is seen that for $p = 1$ it takes $n - 1$ rounds to complete an accumulation at v_0, another $n + 1$ rounds until v_0 sends out its final packet, and a final $(\frac{n}{2} - 2)$ rounds until the last packet sent by v_0 reaches its final destination. Thus, $r_{H_1}(1, B_n) \leq \frac{5n}{2} + O(1)$

For $p = 2$, it takes $\frac{n}{2}$ rounds to complete an accumulation at v_0, $(\frac{n}{2} + 1)$ more rounds until v_0 sends out its last packet, and a final $(\frac{n}{2} - 2)$ rounds until the last packet reaches its final destination. Thus $r_{H_1}(2, B_n) \leq \frac{3n}{2} + O(1)$

Now, let $G_{m,n}$ denote a rectangular grid with m rows and n columns. Since $G_{m,n}$ contains mn vertices it follows from Equation 2.1 that

$$r_{H_1}(p, G_{m,n}) > \frac{2mn}{p} + 1.44 \log_2 p$$

A natural way to gossip in $G_{m,n}$ would be

> Gossip in each of the columns in parallel, and then gossip in each of the rows in parallel. [4]

Such an approach uses $\frac{5mn}{2p}$ rounds. Bagchi, Schmeichel and Hakimi [4] proposed a slightly more complex algorithm taking $\frac{2mn}{p} + O(m+n)$ rounds which is the best known algorithm for the grid developed so far.

For a ring R_n on n vertices the following results have been developed in [4].

$$r_{H_1}(1, R_n) = \begin{cases} 2n-2 & \text{for even } n \\ 2n & \text{for odd } n \end{cases}$$

$$r_{H_1}(p, R_n) \leq \begin{cases} \lceil \frac{n}{2} \rceil + \lceil \frac{n}{p} \rceil + \frac{p}{2} - 1 & \text{for even } p \\ \lceil \frac{n}{2} \rceil + \lceil \frac{n}{p-1} \rceil + \frac{p-1}{2} - 1 & \text{for odd } p \text{ and } p > 1 \end{cases}$$

Algorithms accomplishing the above bounds have also been given in [4].

Let T_n denote the complete binary tree on n vertices. For gossiping in this topology an efficient algorithm has been proposed in [4] which can be described briefly as follows: Color the edges of T_n with three colours $\{0, 1, 2\}$. If $j = (r \bmod 3)$, the edges coloured j will be termed active during round r. Throughout the algorithm each node v will maintain a queue $Q(v)$ which initially contains just v's message. As usual, each node also maintains a list of all messages received in its local memory, which is separate from the queue. As messages arrive at v from v's children, they are inserted in $Q(v)$ in the order received. Let $F(v)$ denote the parent of v in T_n. Suppose $(v, F(v))$ is active during round r. If $Q(v)$ is nonempty, then v sends $f(v)$, the first $\min\{p, |Q(v)|\}$ messages in $Q(v)$ in a packet to $F(v)$. Otherwise, if $F(v)$ knows any messages not yet known to v, then $F(v)$ sends v a packet containing as many messages, not yet known to v, as possible. The algorithm halts when each unit knows all the messages.

It has also been proved that the above algorithm is optimal within $O(1)$ rounds if n is sufficiently large relative to p.

For arbitrary connected graphs, no efficient algorithm for gossiping has been reported so far.

2.3 Telephone Model

The next model that we consider is the telephone model. Because of its similarities with a full duplex communication channel, this model is also known as the F_1 model. The subscript 1 signifying that the maximal number of simultaneous communication partners at each node is 1. Like the mail model this model too is applicable in many practical parallel computing problems.

In this model, in a single round, each node may be active only through one of its adjacent edges and if it is active then it can simultaneously send and receive messages through this active edge. In other words, if an edge is used for communication, the information flow on this edge can be bidirectional. Thus a two-way communication algorithm for a graph $G = (V, E)$ can be described as a sequence $E_1, E_2, \ldots, E_r$ of some sets (matchings) $E_i \subseteq E$, where for each $i \in \{1, 2, \ldots, r\}$, $\forall (x_1, y_1), (x_2, y_2) \in E_i$: $\{x_1, y_1\} \neq \{x_2, y_2\}$ implies $x_1 \neq x_2 \wedge x_1 \neq y_2 \wedge y_1 \neq y_2 \wedge y_1 \neq x_2$.

To explain the above concepts more clearly, a gossip algorithm for a ring of 4 nodes is shown in Figure 2.2.

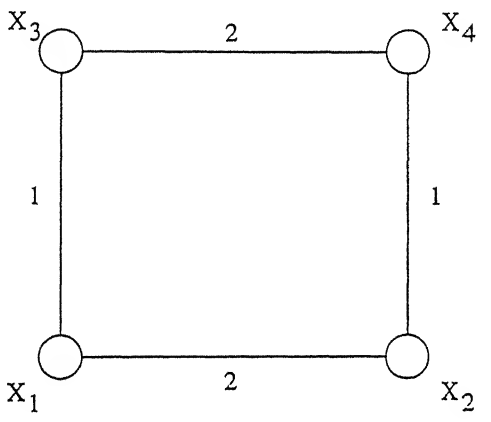


Figure 2.2: Gossip algorithm for a ring of 4 nodes in the Telephone model

In the first round nodes x_1, x_3 and nodes x_2, x_4 exchange the information they have in a manner equivalent to a telephone call between the communicating nodes. In the second round, nodes x_1, x_2 and nodes x_3, x_4 exchange their accumulated knowledge upto that point. Thus it takes two rounds to complete the gossip. The communication algorithm in this case can be described as: $\{(x_1, x_3), (x_2, x_4)\}, \{(x_1, x_2), (x_3, x_4)\}$ and it can be verified that all the properties of the telephone model are satisfied .

In the next few paragraphs, some widely used gossip algorithms for this communication model are discussed.

2.3.1 Unbounded Information Exchange in a Communication Step

Once, again the following rather discouraging result is available but because of great significance of this model in parallel computing applications various bounds and algorithms for certain special classes of graphs have been obtained.

> The problem of finding time optimal algorithms for gossiping in general graphs is NP-hard under this model of communication [15]

In this context, let us denote by $r_{F_1}(G)$ the number of rounds required to complete gossiping in an n-node graph $G = (V, E)$.

For a complete graph K_n, the value of $r_{F_1}(K_n)$ has been obtained independently by many people, e.g., Bavelas [5], Landau [39], and Knodel [36] and is:

$$t(K_n) = \begin{cases} \lceil \log_2 n \rceil & \text{if } n \text{ is even} \\ \lceil \log_2 n \rceil + 1 & \text{if } n \text{ is odd} \end{cases}$$

Gossip algorithms achieving this bound can be found in [27].

In 1986, Labahn [38] investigated the time required to gossip in a tree network. If $r_{F_1}(T)$ be the number of rounds required to gossip in a tree, T of n vertices, then Labahn showed that

$$2\lceil \log n \rceil - 1 \leq r_{F_1}(T) \leq 2n - 3$$

He also characterized the trees which achieve gossiping in $2\lceil \log n \rceil - 1$ rounds and described how to construct these trees and their minimum time calling schemes.

For a hypercube of n nodes, the number of rounds required to gossip under the F_1 model has been proved to be the same as that required to broadcast from a single node i.e., $\lceil \log n \rceil$ if n is odd and $1 + \lceil \log n \rceil$ if n is even [27].

For rings, two-dimensional grids, toroidal grids and Illiac grids, Farley and Proskurowski [23] have carried out an extensive analysis of the gossip problem. Their results are summarized as follows. For the n-node ring the minimum number of rounds to gossip is $\frac{n}{2}$ (the diameter) if n is even and $\lceil \frac{n}{2} \rceil + 1$ if n is odd (two more than the diameter). The minimum for any two-dimensional grid other than the 3×3 grid is the diameter of the grid; for the 3×3 grid it is 5 (one more than the diameter). The minimum number of

rounds to gossip for an $N \times M$ toroidal grid is equal to the diameter, if N and M are even, is between 1 and 2 greater than the diameter if just one of N and M is odd, and is between 2 and 4 greater than the diameter if both are odd. Similar results are obtained for the Illiac grid. For d-dimensional grids and toroidal grids in general, the results in [23] indicate that the minimum number of rounds to gossip is no greater than $2d$ plus the diameter. Whether it is possible to do better than that is an open question; for example it is not known whether the $3 \times 3 \times 3$ grid can be solved in six steps (rounds) under the F_1 model.

For a bus B_n consisting of n nodes, $x_1, x_2, \ldots, x_n$, gossip algorithms working in $n-1$ rounds for even n and n rounds for odd n can be found in [23]. A simple algorithm working in $n-1$ rounds is $A = E_1, E_2, \ldots, E_{n-1}$, where $E_i = \{(x_1, x_2), (x_3, x_4), \ldots, (x_{n-1}, x_n)\}$ for odd i, and $E_j = E(B_n) - E_1$ for all even j. Similar algorithms can be given for any odd integer $n \geq 3$.

2.3.2 Bounded Information Exchange in a Communication Step

The computational complexity of determining an algorithm to complete gossip in an optimal number of rounds for general graphs has not been investigated for this model, but it is very likely that it is NP-hard because it is known to be NP-hard when there is no bound on the information exchanged in a communication step. However some work has been done by Bermond et al [6] which is summarized below. Firstly, let us consider the extremal case in which gossiping is to be performed under the restriction that the communicating nodes can exchange exactly one message in each round

Let $r_{F_1}(1, G)$ denote the minimum possible number of rounds required to complete gossiping under this model in a graph G with the condition that in each round the communicating nodes can exactly exchange one message per packet. For any graph $G = (V, E)$ with $|V| = n$, a lower bound on $r_{F_1}(1, G)$ has been given in [6] which is nonetheless quite useful. It is as follows: If $m(G)$ is the size of a maximum matching in G. then

$$g(1, G) \geq \lceil \frac{n(n-1)}{2m(G)} \rceil$$

For any graph $G = (V, E)$, the size of a maximum matching is at most $\lfloor |V|/2 \rfloor$. Hence, from the above inequality it follows that, the gossiping time $r_{F_1}(1, G)$ of any graph G

with n nodes is always lower bounded by

$$r_{F_1}(1,G) \leq \begin{cases} n-1 & \text{for even values of } n \\ n & \text{otherwise} \end{cases}$$

This lower bound is actually attained in Hamiltonian graphs.

For a ring R_n of n nodes Bermond [6] and others have given algorithms to gossip in $n-1$ rounds for even n and n rounds for odd n. We now give a brief description of the algorithm for even n. The case for odd values of n is similar.

For each integer t define

$$M_t = \begin{cases} \{(v,w) : v \text{ is even and } w = v+1\} & \text{if } t \text{ is even} \\ \{(v,w) : v \text{ is odd and } w = v+1 (\mathrm{mod} n)\} & \text{if } t \text{ is odd} \end{cases}$$

The gossiping algorithm to be followed is as given below

Round $t = 1$: Each node v sends its own message to a node w for which (v,w) belongs to M_1

Round $t = 2$: Each node v sends its own message to a node w for which (v,w) belongs to M_2

Round t, $3 \leq t \leq n-1$: For each node v let w be a node such that (v,w) belongs to M_t. Node v sends a new message to w, i.e., v sends the message it has received first among those that v has neither received from w nor sent to w in a previous round.

Following this algorithm, for even values of n, the task of gossiping can be completed in $n-1$ rounds.

For a bus B_n of n nodes and any k-ary tree $T_{k,n}$ on n nodes, the following results have been established.

$$r_{F_1}(1,B_n) = \begin{cases} 1 & \text{if } n = 2 \\ 4 & \text{if } n = 3 \\ 2n-3 & \text{if } n \geq 4 \end{cases}$$

$$r_{F_1}(1,T_k,n) = \begin{cases} k^2 & \text{if } n = k+1 \\ 2k^2+k+1 & \text{if } n = 2k+1 \\ (k+1)(n-1)-k & \text{if } n \geq 3k+1 \end{cases}$$

The corresponding algorithms can be found in [6].

For any Hamiltonian graph G_H on n vertices the following results are known.

$$r_{F_1}(1, G_H) = \begin{cases} n-1 & \text{if } n \text{ is even} \\ n & \text{if } n \text{ is odd} \end{cases}$$

Let us now study the case, where gossiping has to be performed under the restriction that in each round (more specifically at each time instant), communicating nodes can exchange upto p messages in a packet where p is fixed but arbitrary otherwise. It is assumed that p is smaller than the number of nodes in the graph G otherwise the problem is equivalent to the previous one considering unbounded length transmissions. Again let $r_{F_1}(p, G)$ denote the minimum number of rounds required to accomplish gossiping in this case. Some lower bounds on $r_{F_1}(p, G)$ based on elementary counting arguments are given below [6].

For any graph $G = (V, E)$, with $|V| = n$, and integer p such that $2^{\lceil \log p \rceil} \leq n$ we have

$$r_{F_1}(p, G) \geq \begin{cases} \lceil \log p \rceil + \lceil \frac{n - 2^{\lceil \log p \rceil}}{p} \rceil & \text{if n is even} \\ \lceil \log p \rceil + \lceil \frac{n - 2^{\lceil \log p \rceil}}{p} \rceil + 1 & \text{if n is odd} \end{cases}$$

In a complete graph K_n with n nodes following results exist. For each even integer n and integer p such that $2^{\lceil \log p \rceil} \leq n$,

$$r_{F_1}(p, K_n) = \lceil \log p \rceil + \lceil \frac{n - 2^{\lceil \log p \rceil}}{p} \rceil$$

For each odd integer n and integer p such that $2^{\lceil \log p \rceil} \leq n + 1$,

$$\lceil \log p \rceil + \lceil \frac{n - 2^{\lceil \log p \rceil}}{p} \rceil + 1 \leq r_{F_1}(p, K_n) \leq \lceil \log p \rceil + \lceil \frac{n + 1 - 2^{\lceil \log p \rceil}}{p} \rceil + 2$$

If p is a multiple of 4 then

$$r_{F_1}(p, K_n) = \lceil \log p \rceil + \lceil \frac{n - 2^{\lceil \log p \rceil}}{p} \rceil + 1$$

For a d-dimensional hypercube H_d with 2^d nodes Bermond and others [6] proved the following result: For each integer $p < 2^d$,

$$r_{F_1}(p, H_d) = \lceil \log p \rceil + \lceil \frac{2^d - 2^{\lceil \log p \rceil}}{p} \rceil$$

For a ring R_n and bus B_n with n nodes, the following results have been established. For each $n \geq 3$ and $p \geq 2$,

$$r_{F_1}(p, R_n) = r_{F_1}(2, R_n) = \begin{cases} n/2 & \text{if } n \text{ is even} \\ (n+3)/2 & \text{otherwise} \end{cases}$$

For each $n \geq 2$ and $p \geq 2$

$$r_{F_1}(p, B_n) = r_{F_1}(2, B_n) = 2\lceil \frac{n}{2} \rceil - 1$$

For general graphs no well known algorithm for gossiping has been given so far.

2.4 Full-Duplex, Delta-port Model (or the Shouting Model)

Another important model used extensively in various parallel computing applications is the full duplex, delta port model. It is also known as the $F_\star$ model - F signifying the full duplex nature and the subscript $\star$ signifying that communication with any number of neighbouring nodes is allowed.

The earliest work done for this model was by Gentleman [29], who studied the communication complexity of matrix computations on grid interconnection networks like that in ILLIAC IV. Gentleman used this $F_\star$ model of communication and showed, for example, that at least $0.70n$ communication steps were needed to invert an $n \times n$ matrix stored in an $n \times n$ grid. In [26] Fragopolou has given several communication and fault tolerant gossip algorithms on certain interconnection networks for this model.

In this model, during each round each node can communicate with all its neighbours. More exactly it can send a packet to *all* its neighbours *and* receive a packet from all its neighbours. Furthermore, each node is allowed to send different packets to different neighbours in a round.

Let us now take a look at some of the gossip algorithms for this model of communication. Let us denote by $r_{F_\star}(p, G)$ the minimum number of rounds required to complete the gossiping process in a graph G in this communication model - p here denotes the number of messages the communicating nodes are allowed to exchange during a round. For unbounded length transmissions $p = \infty$.

2.4.1 Unbounded Information Exchange in a Communication Step

Under the $F_\star$ model, the number of rounds to gossip in any graph, G, has the graph's diameter, $D(G)$, as a tight bound. So, if we have no restriction on the number of messages in a packet communicated between the nodes, then in the $F_\star$ model, we can gossip in $D(G)$ rounds. It is sufficient to use the greedy protocol where at each round each node sends all the messages it has just received during the preceding round to all its neighbours. Hence in this case

$$r_{F_\star}(\infty, G) = D(G) \tag{2.2}$$

2.4.2 Bounded Information Exchange in a Communication Step

For this model although the computational complexity of computing $r_{F_\star}(1, G)$ or $r_{F_\star}(p, G)$ for a fixed p, has not been determined, it is very likely that it will be NP-hard. Some bounds have been given in [7], which are as follows: For a graph G having diameter $D(G)$,

$$r_{F_\star}(1, G) \geq D(G)$$

For a graph G having n vertices and minimum degree d,

$$r_{F_\star}(1, G) \geq \lceil \frac{n-1}{d} \rceil \tag{2.3}$$

For a digraph G with n vertices and minimum in-degree d,

$$r_{F_\star}(1, G) \geq \lceil \frac{n-1}{d} \rceil$$

For Cayley symmetric digraphs having a complete rotation an optimal algorithm has also been proposed which achieves the lower bound given by Equation 2.3. This follows from a more general property valid for Cayley symmetric digraphs with special automorphisms. This leads to optimal algorithms for k-dimensional tori, hypercubes and star graphs as these can be considered as special cases of Cayley symmetric digraphs. The techniques developed in [7] can easily be extended to the case where p messages per packet are allowed to be sent (and/or received) during each round and it can be shown that for the networks considered in the above reference, one can optimally gossip in $\lceil \frac{n-1}{pd} \rceil$ rounds.

A related communication model is the half-duplex, delta port model also known as the $H_\star$ model of communication. In this model it is assumed that during each round, each node can communicate with all its neighbours. However it can either send or receive messages unlike the $F_\star$ model in which both can be performed simultaneously in a round. The model of communication for distributed memory architectures can be considered equivalent to this model of communication. Let us now take a look at some of the known gossiping results for this communication model. All the results given below are for unbounded number of messages in a packet.

For any bipartite graph with diameter D under the $H_\star$ model, there is a $(D+1)$ - round solution to the gossip problem as follows: two-colour the nodes, red and black, and have red nodes transmit to all neighbours in even rounds and black nodes do so in odd rounds.

For the complete graph, it takes exactly two rounds to solve the problem regardless of the size of the graph. It is impossible to solve the problem in one round, and for a two-round solution we just select a subgraph consisting of one node and one edge from it to every other node and apply the above construction with the first transmissions going towards the central node.

Since the d-dimensional hypercube is bipartite, the above bipartite approach yields a $(d+1)$-round solution. For $d \leq 2$ this is optimal, but for $d \geq 3$, solutions in d rounds exist. They are hard to find and describe. For example, there are 2^{36} different three-round strategies to consider on a 3-cube. This is described in more detail in [37]. One significant result given in [37] is that for all values of $d \geq 4$, time-invariant optimal solutions, that is, solutions where the transmissions are the same in each round, exist.

For the d-dimensional grid of diameter D, the problem can be solved in D rounds even under the pairwise model, and this automatically provides an optimal solution under the simultaneous model.

A ring with an even number of vertices N is bipartite, and the above bipartite algorithm gives an optimal solution in $(\frac{N}{2}+1)$ rounds that is one more than the diameter. (It is impossible to do it in $\frac{N}{2}$ rounds). With an odd number of vertices, the following modified version can be used. In each round, one link is omitted from involvement, the choice of which link to omit proceeding around the ring one step at a time. The other

links participate in the basic bipartite pattern, where directions are chosen so that each node alternates sending and receiving. This yields a solution requiring $\frac{(N-1)}{2} + 2$ rounds, two more than the diameter.

For the d-dimensional toroidal grid the same construction can be used. In each dimension where the size is odd, the pattern must be modified as with the ring, where the omitted links are chosen with matching coordinates so that they form entire rows or columns (or hyperplanes). This yields solutions that take $(D + k + 1)$ rounds, where k is the number of dimensions where the size is odd.

For transmissions of bounded length under the $H_\star$ model, as far as we know no results have been published so far.

2.5 Possible Generalisations of Above Models

Apart from the communication models discussed in the previous sections, other popular models also exist. However most of them can be considered as generalizations of those that we have discussed above.

One way to generalize the one-way communication model (or telegraph model) is to allow more activities for an active node in one round. (Note that an active node in the one-way communication round is either a sender or a receiver. For example, (i, j)-model can be constructed in which in any round one node can send a message to i neighbours via i adjacent edges and it can receive messages from j neighbours via j adjacent edges. Thus the two way model (also called the Telephone Model) is a restricted version of the (1.1)-model, where additionally, any active node must use the same adjacent edge for both submission and reception. The (i, j)-model with several possible additional restrictions provides a rich variety of communication models for further investigation.

Another possibility to generalize the one-way (two-way) model is to consider the rounds of one-way communication algorithms as sets of vertex-disjoint paths of length 1 instead of as sets of directed edges (which is clearly equivalent). The generalization consists of allowing an arbitrary length of these paths in each round. Thus, a round is described by a set of vertex-disjoint paths, where additionally a direction from one end-point to the second end-point may be prescribed for any path. What can happen on these

paths in one round is determined by the communication model. Obviously, this provides several possibilities. The two possibilities used in the literature [22, 24] considering either that one end-node broadcasts its whole knowledge to all other nodes of the given path or that one of the end-nodes sends its knowledge to the other end-node and the remaining nodes on the path do not read the packet submitted. For the first possibility several optimal communication algorithms [24] have been constructed. These models are known as vertex-disjoint models and one possibility to generalize them is to define the so-called edge-disjoint models where each round is described by a set of edge-disjoint paths and what can happen on one path in one round can be determined in different ways

Similarly, one can introduce a variety of further communication models based on generalizations of the other existing models.

2.6 Radio Network Model

All the communication models discussed till now have relevance to various multiprocessor interconnection schemes and parallel computing problems where the efficiency of an algorithm for gossiping on a particular topology is a useful measure of that graph's communication capability when used as an interconnection network. Because of this, most of the gossip algorithms developed for the above models have been for special classes of graphs, specifically those that can be used as interconnection networks in a multiprocessor arrangement. Most wireline networks can be modelled only as a general graph and thus some of the gossip algorithms developed for the telephone model with a bounded number of messages per packet can be used. However, very few efficient algorithms for general graphs in the telephone model are available, primarily because this model has been studied more from the point of view of its applications in multiprocessor interconnection networks than in other generalised communication applications.

Radio networks, due to certain of its peculiar properties like being a broadcast communication medium and associated interference constraints, cannot be modelled via any of the communication models considered above. Here, we need a model that takes into consideration the special properties of a radio network environment. This has motivated the definition of a new model which we refer to as the "Radio Network Model". In the

radio network model each node is assumed to be equipped with a radio transmitter and a radio receiver. When a node transmits, all nodes within the range of the transmitter can receive the transmission correctly if there are no simultaneous transmissions received at the receiver. Simultaneous transmissions result in collisions in the area covered by the ranges of one or more of the simultaneous transmitters where all information contained in the packet is lost.

For construction and analysis of our algorithms we present the following network model. A multihop radio network is represented by a graph $G = (V, E)$. Vertices in V represent the network nodes, $|V| \equiv N$ and any undirected edge is interpreted as two antiparallel directed edges. A directed edge $u \rightarrow v$ signifies that v receives every signal (packet) transmitted by u, whereby u is defined as an incoming neighbor of v. A correct (collision free) reception of a message by v is conditioned by the following.

- v does not transmit in the same slot u does, i.e., v cannot transmit and receive simultaneously.
- u is the only incoming neighbor of v transmitting in the same slot, i.e., v cannot receive two simultaneously incoming transmissions.

Violations of the above two conditions define the two cases of message collisions in the radio network.

Very little work has been done on gossiping in radio network. The only work that we are aware of is that by Ravishankar, Singh and Sridhar [46, 47, 51]. In a series of papers by them they worked on building algorithms for nodes connected as rings or buses, allowing packets that can contain an unbounded number of messages, clearly an unrealistic assumption for real networks. We are not aware of any work that has been done for gossiping with bounded number of messages per packet in radio networks.

2.6.1 Unbounded Information Exchange in a Communication Step

The first work on gossiping done for this model is the work done in [51]. Here an asymptotically optimal gossiping algorithm for the case where the nodes are placed on a line of length L is given. The gossiping algorithm proposed proceeds in three distinct stages.

The basic idea is to first allow as many nodes as possible to transmit simultaneously so that all the messages are made known to a smaller set of nodes (taken together). A broadcast is then initiated from both ends of the configuration towards the middle. As these broadcasts proceeds every node transmits all the messages it knows. Thus eventually, some set of nodes are the first to hear both the broadcasts and know all the messages. This set of nodes do not forward the two broadcasts but instead initiate a new broadcast simultaneously in both directions (outward from the node) containing all the messages.

In [46], Ravishankar and Singh proposed a distributed algorithm for gossiping when the nodes are placed randomly on a ring and again the assumption made is that during a transmission a node can transmit all the messages that it has collected thus far in a single packet (communication step). The algorithm proposed proceeds in two phases. During the first phase, the idea is to allow all nodes to transmit their messages at least once so that a smaller set of nodes collectively know all the messages. To do this efficiently the algorithm allows as many nodes as possible to transmit simultaneously. In the second phase, broadcasts are initiated in both directions. At any time there are broadcasts proceeding in the clockwise direction as well as in the anticlockwise direction. When any two broadcasts moving in the opposite directions cross one another it results in a collision. These broadcasts collect all messages as they proceed. When all these broadcasts terminate, gossiping is complete.

2.6.2 Bounded Information Exchange in a Communication Step

As far as we know no algorithms for gossiping have been given for this case. We have proposed some algorithms in this thesis for gossiping in this situation which can be applied to radio networks modelled as general graphs. We expect that gossiping in radio networks will be overwhelmingly used in control and monitoring of networks which are typically modelled as general graphs. Therefore, we claim that efficient gossip algorithms are more useful for the radio network model on general graphs rather than on restricted classes of graphs. In addition we have also proposed some algorithms for gossiping in some restricted network topologies. The topologies are chosen from the point of view of their applications in personal communication systems. We believe that the proposed algorithms would definitely find their use in existing packet radio networks where gossiping

could be used to exchange control information among nodes. Moreover the algorithms can serve as a benchmark for other future work which can be done in this field.

Chapter 3

Scheduling in Radio Networks

3.1 Introduction

In multihop packet radio networks the path from a source to a destination may involve many intermediate nodes (hops). These intermediate nodes receive a transmitted packet from the node of the previous hop and forward it on the path towards the destination to the node on the next hop. The communication channel, which must be shared by all the network users, is the critical system resource and should be used efficiently. Efficient use would mean share or reuse this resource as much as possible. Channel sharing can be done in three ways - frequency, time and space. Frequency reuse is essentially through frequency division multiple access (FDMA) or modern approaches such as code division multiple access (CDMA) and spread spectrum [53] can be used. Time reuse is usually defined by the channel access protocol. Spatial reuse (widely used in multihop radio networks) means that frequency used by a node in some part of the network is also used in another part where the signal from the first node cannot cause interference at the intended receivers. This is possible because of propagation loss or because of the fact that one part of the network may be shielded from other parts. In this thesis we are concerned with packet radio networks that are assigned a single frequency. A given communication objective can thus be achieved efficiently (in time) by reusing the broadcast channel in space and time. This possibility of channel reuse demands the need for scheduling in multihop radio networks. Thus, in a multihop packet radio network, since a single channel is shared by all the nodes, the transmission for each node must be scheduled to

avoid collisions or interference that cannot be allowed by the given application.

A multi-access protocol, i.e., a spatial time-division multiple-access (TDMA) protocol can be used to schedule conflict-free transmissions. In this spatial TDMA network time is divided into frames which consist of fixed length time slots. When certain nodes transmit simultaneously, collisions which cannot be tolerated will occur. Therefore, any two nodes, whose simultaneous transmissions may result in a collision (or interference) must be scheduled to transmit at different time slots, while nodes some distance away (those that may not be affected) may be scheduled to transmit in the same time slot. The scheduling algorithm should yield a schedule that achieves the communication objective as efficiently as possible.

The input to a scheduling algorithm is a graph $G = (V, E)$ representing the connectivity of the nodes in the network and the output is the schedule for a frame F. The frame will consist of l slots and the schedule determines the transmitters in each slot. The schedule for the frame, F is thus a set $\{s_1, \ldots, s_l\}$ consisting of l elements corresponding to the l slots of the frame. In slot t of the frame, $1 \leq t \leq l$, the nodes belonging to the set $s_t \subset V$ are allowed to transmit while all other nodes are directed to remain silent. The nodes constituting the set s_t and the messages they are to transmit in this time slot t is decided from the requirements of the communication objective.

3.2 Graph Colouring : A Paradigm for Scheduling

The problems of vertex and edge colouring in graphs are well known and have been extensively studied for decades. See for example [20] for some reviews. A vertex colouring of a graph G is an assignment of 'colours' to the vertices of a graph so that no two adjacent vertices have the same colour. We define the chromatic number of a graph G, $X(G)$, to be the minimum number of colours needed to obtain a vertex colouring of G. If $X(G) = k$, then G is k-vertex-chromatic and if $X(G) \leq k$, then G is k-vertex-colourable.

An edge colouring is defined analogously as an assignment of 'colours' to the edges of a graph so that no two adjacent edges have the same colour. The chromatic index of G (as distinct from chromatic number for vertex colouring), $X'(G)$, is defined to be the minimum number of colours needed to obtain an edge colouring of G. If $X'(G) = k$, then

G is k-edge-chromatic. If $X'(G) \leq k$, then G is k-edge-colourable.

Graph colouring has served as a paradigm for several applications such as production scheduling, timetabling etc. and has been extensively studied. Though most colouring problems are conceptually simple, they are notorious for their difficulty. One may be aware of the Four Colour Conjecture that every planar graph is four-colourable. This was an open problem for over a century, before it was proved in 1976.

While many applications can be modelled as vertex or edge colouring problems, the problem of scheduling in radio networks can be modelled as a modified version of the above problems. In this colouring paradigm for scheduling, a slot is represented as a colour. Scheduling the transmission from a node (along an edge) in slot k is equivalent to the colouring of the corresponding node (edge) with colour k. Let us now explain this paradigm more clearly with respect to the problem of broadcast scheduling. Recall that in broadcast scheduling, the transmission of a node must be received correctly by all of its neighbouring nodes. Therefore, not only must the neighbours of a node u be silent when u is transmitting but also must the neighbours of the neighbours of u be silent at this time. This is to allow the neighbours of u to receive the transmission of u without interference. Clearly, the same time slot t cannot be assigned to nodes that are within two hops of each other. In the context of colouring, where a slot k corresponds to the colour k, the problem is clearly equivalent to one of colouring the graph representing the radio network so that no two vertices that are adjacent or distance-2 apart receive the same colour. Other scheduling problems can also be converted to graph colouring methods in the same manner. In the next few sections we will discuss this correspondence in more detail taking into consideration some specific scheduling problems.

3.3 Broadcast Scheduling

In a broadcast scheduling problem the entities to be scheduled are the nodes themselves. The transmission of a node is intended for and must be received collision free by all of its out-neighbours. Thus, two nodes may not be assigned to the same slot if either they are adjacent or have a common neighbour. Thus, broadcast scheduling is one of colouring the vertices of the graph such that any pair of vertices (u, v) may be coloured the same

if and only if:

- edge $(u, v) \notin E$, and
- there is no w such that $(v, w) \in E$ and $(u, w) \in E$.

When the first condition fails to hold, there is a primary vertex conflict between vertices u and v, and when the second condition fails to hold there is a secondary vertex conflict between them. The vertices in conflict with any vertex $v \in V$ are those which are either adjacent or distance-2 apart.

The problem of broadcast scheduling is closely related to the classical graph theoretic problem of vertex colouring. More specifically, in a broadcast schedule problem no two nodes that are adjacent or distance-2 apart should receive the same colour whereas in a vertex colouring problem adjacent nodes should be assigned different colours. This problem of broadcast scheduling is more specifically known as the distance-2 colouring problem. The vertex colouring problem is a notoriously hard problem for which the best existing approximation algorithms are quite poor and for which it is widely believed that there does not exist approximation algorithms with good performance bounds. Therefore it is believed that the distance-2 colouring problem is an equally hard problem. In fact it has been shown in [44] that the problem of finding an optimal schedule, that is a minimum length schedule, is NP-complete for broadcast scheduling. However, for certain specific architectures like star, bus and ring networks more or less optimal algorithms have been proposed.

A star network consists of a 'central' node connected to several other nodes. Each of the non-central nodes can communicate only with the central node. This is shown in Figure 3.1(a). For a star network consisting of N nodes and a maximum vertex degree of ρ distance-2 colouring requires at least $\rho + 1$ colours, since each of the vertices are within distance-2 of each other. So, an optimal colouring of a star is trivially obtained by assigning a different colour to each vertex.

A bus network consists of nodes connected in a 'straight-line' as shown in Figure 3.1(c). Each node can communicate with at most two neighbouring nodes. For a bus, using colours 1, 2 and 3 in a round-robin fashion from left to right $(1, 2, 3, 1, 2, 3, 1 \ldots)$ will yield an optimal broadcast schedule giving a frame consisting of just three time slots.

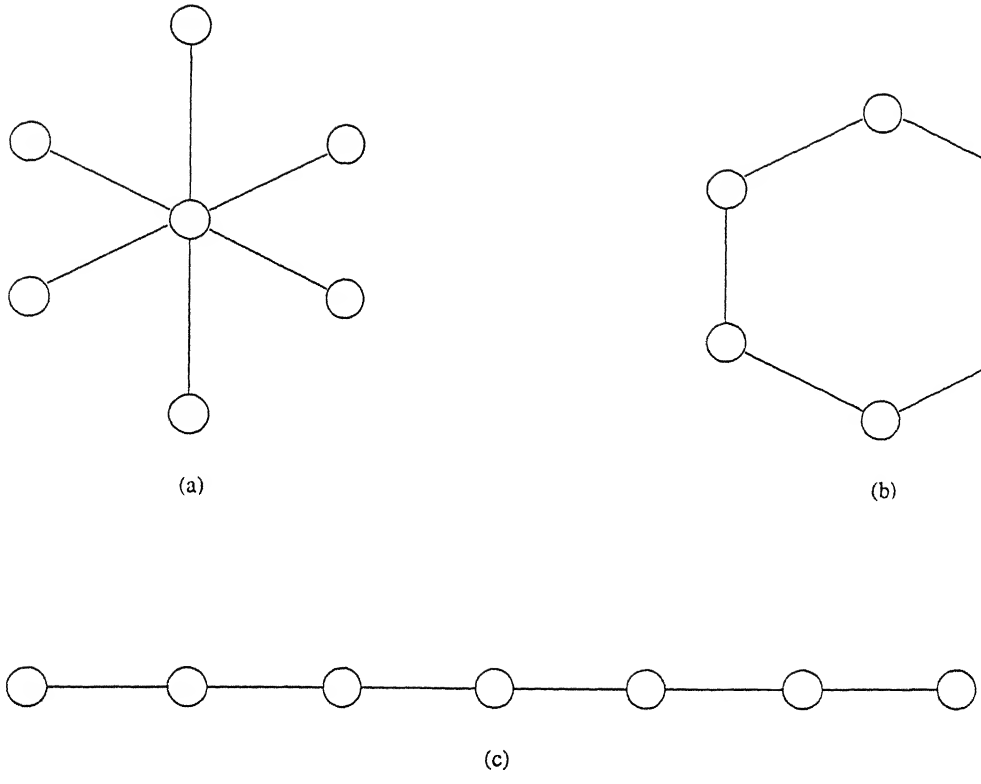


Figure 3.1: (a) Star, (b) Ring and (c) Bus Networks

A ring network consists of nodes connected in a ring as shown in Figure 3.1(b). Thus a ring network is basically a closed bus network. A ring with $3m$ vertices can be coloured using exactly three colours, say 1,2 and 3 and the colouring procedure consists of beginning from any node in the ring and colouring in a round-robin fashion till the starting node is reached. For a ring with $3m + k$ vertices ($k = 1$ or 2), the colouring operation is identical to the $3m$ case until the $(3m)^{\text{th}}$ vertex. Thus, exactly three colours will be used up to this point. After this the remaining k vertices can be coloured with completely different colours thus requiring a maximum of $k + 3$ colours.

For arbitrarily connected topologies, since the problem has been shown to be in NP, various good approximation algorithms have been proposed in literature. A centralised and distributed algorithm for a multihop packet radio network is provided in [17]. In this scheduling algorithm, a network wide 'skeleton' schedule is first constructed. The schedule is then 'filled' in a distributed manner. The schedule produced is maximal in that the addition of a node s to a slot to which it is not originally assigned will cause a conflict in the schedule. No analysis of the performance is presented. In [44] a centralised

greedy algorithm is presented. This is a straightforward algorithm in which a node is chosen at random and coloured with the first available, nonconflicting colour (least colour not assigned to any of the conflicting vertices). Another algorithm similar to this pure greedy algorithm is the maximum degree first algorithm which selects the vertices in decreasing order of their degrees for colouring. The colouring is done "greedily", that is, by using the least colour not used by one-hop or two-hop neighbours. In [14] the problem of selecting a set of 'ready' nodes to transmit in the current slot so that the number of successful transmissions for that slot will be maximum is considered. The issue of fairness in broadcast scheduling is discussed in some detail in [13] wherein two fair heuristic algorithms are described for a problem similar to the broadcast scheduling problem. The performance is analyzed using simulations. One of the algorithms developed by us also uses a broadcast schedule to complete the task of gossiping and is discussed in detail in Chapter 4.

3.4 Link Scheduling

In a link scheduling problem the links between nodes are scheduled. The transmission of a node is intended for a particular out-neighbour, and we require that there be no collision at this receiver. Thus, two links may not be assigned to the same slot if either they are adjacent or there exists a third link from the transmitter of one link to the receiver of the other link. We assume in this section that the graph representing the network is a directed graph $G = (V, E)$. An undirected graph can be considered as a special case of a directed graph in which every $(a, b) \in E$ implies that (b, a) also belongs to E.

In contrast to broadcast scheduling, link scheduling only requires that the transmission of a node be received by the intended receiver (assumed to be exactly one of the neighbours). In such a case, the (directed) links between the nodes are the entities that must be assigned slots. In the graph-colouring context, this involves the assignment of colours to the directed edges of the graph such that the corresponding schedule is conflict free. Recall that the schedules are constrained by the transmission-reception and the interference constraints. The transmission-reception constraint requires adjacent edges to

receive different colours (as in the conventional edge-colouring problem). The interference constraint additionally requires that two edges e_1 and e_2 be coloured differently if there exists an edge from the source of e_1 to the destination of e_2. Such a colouring is known as the 'prn colouring' problem (prn comes from the abbreviation for packet radio network). Thus, link scheduling corresponds to colouring the edges of a directed graph $G = (V, E)$ such that any pair of directed edges (a, b) and (c, d) may be coloured the same if and only if:

- a, b, c, d are all mutually distinct, and
- $(a, d) \notin E$ and $(c, b) \notin E$.

Naturally, we are interested in finding out if an algorithm that produces a minimum length link schedule frame exists. Again, just like for the broadcast scheduling problem, it has been shown in [18, 2] that the optimization version of the link scheduling problem is NP-complete. In other words, the complexity of an algorithm that produces a minimum length frame is likely to be exponential in the size of the number of nodes in the network. Thus the only practically feasible algorithms that produce an optimal schedule are restricted to small networks. Again, more or less optimal algorithms have been proposed for certain specific architectures like star, bus and ring topologies.

An optimal prn-colouring of a star network is trivially obtained by assigning a different colour to each of its edges. For a path an optimal link scheduling algorithm is given in [40]. In [18] an optimal algorithm for the prn-colouring of ring networks is provided. The algorithm described there examines several different cases and gives an explicit colouring for each of these cases.

For arbitrarily connected packet radio networks, a considerable amount of literature exists on the problem of link scheduling. One of the earliest papers is by Silvester [50] in which the problem of link scheduling is modelled as one of colouring the vertices of a 'derived' graph. Every vertex in the derived graph corresponds to an edge in the original graph and two vertices are adjacent if and only if the corresponding edges are interfering. The vertex colouring of this derived graph is done in a greedy fashion by considering the vertices in decreasing order of their degrees. This corresponds to an assignment of the edges of the original graphs to a slot in a conflict free manner. In

[42], a link allocation protocol is proposed. This protocol simulates the links found in a traditional network and therefore protocols designed for the traditional networks may be adapted to the case of radio networks. The main disadvantage of this algorithm is that it does not take into account the broadcast nature of the medium. A link scheduling algorithm for mobile networks is presented in [11]. An analysis of the performance is also given and it is shown to be bounded by the square of the maximum degree. This algorithm is distributed and adapts to topological changes while maintaining the collision free properties. Another algorithm that adapts to topological changes is given in [43]. The time taken for reorganization is proportional to the number of nodes; data traffic may continue whilst the reorganization is taking place. In [12] fairness is considered and is defined in terms of the closeness of the allocation to respective link allocation demands while preserving the collision free property. They give a distributed algorithm for producing a fair scheduling. Most of the algorithms described above use the same basic idea as in the pure-greedy algorithm. Informally the pure-greedy algorithm for prn-colouring picks an uncoloured edge at random and assigns to it the first available colour (that is the least colour not assigned to any of the conflicting edges). A closely related algorithm is the extended maximum degree first ordering (EMDF) algorithm. In this algorithm, we take a maximal mutually conflicting clique of edges around the maximum degree vertex first, colour it, and then progressively do the same for the remainder of the graph. This method is based on the intuitive notion that it is better to colour the more 'crowded' areas first. Another popular algorithm is the arborical link scheduling algorithm [40]. The key to a good performance of this algorithm is the recognition of the fact that while the graph considered as a whole is difficult to colour, it can be decomposed into several pieces each of which is easy to colour. These pieces can then be recombined to produce a good (though not necessarily optimal) colouring of the entire graph. The approach used in this algorithm is to colour each decomposed piece taking into account the influence of the other pieces - this is somewhere in between the approaches of colouring in one 'global' sweep and colouring purely 'locally'.

3.5 Special Purpose Scheduling

In addition to broadcast and link schedules there may be special purpose schedules designed for certain specific applications like for example broadcasting, multicasting, gossiping etc. Every application demands certain criterion and the scheduling developed for the application should cleverly utilize these criterion to frame certain rules for computing the schedule. For example, in the broadcast scheduling problem, the criteria is that a node's transmission should be received collision free by all its neighbours So, the rule for computing the broadcast schedule which should be obeyed by any scheduling algorithm will be that nodes having a distance less than 3 should not have the same colour, so that they do not transmit simultaneously in a slot. In this context, let us briefly explain a special purpose scheduling algorithm developed to complete the task of gossiping in a radio network. This is discussed in more detail in Chapter 4.

The algorithm first constructs a broadcast spanning tree starting from a particular node in the network designated as root. Then the problem of scheduling is to colour the undirected graph $G = (V, E)$, corresponding to the network on which gossiping is to be done, such that the colour assignment for a node $u \in V$ is subjected to the following three constraints

- u's colour must be different from u's parent colour.
- u's colour must not be the colour of any of neighbour of u's parent.
- u's colour must be different from any of u's neighbour's parent.

The order of colouring of the nodes and the parent-child relationship is decided by the order of insertion of nodes in the broadcast spanning tree. Colouring the graph in the above manner yields a frame consisting of a number of time slots, each slot corresponding to a colour used in the colouring process. This frame can be iterated in time to produce a gossip frame in which gossiping can be completed. The number of slots in the frames inside a gossip frame can possibly be further minimized.

It is to be noted that in contrast to broadcast scheduling, implementation of the above gossip schedule would result in certain collisions in the network. That is, it is not guaranteed that in this schedule all the neighbours of a transmitting node would receive

the transmission. The only thing that the schedule is taking care of is that when a node transmits in its scheduled slot then it's parent and children are definitely going to receive the transmission.

This is one example of a special purpose schedule developed for carrying out a specific task, in this case gossiping, in radio networks. In a similar manner schedules can be constructed for other generalized applications.

3.6 Requirements of a Good Scheduling Algorithm

Any scheduling algorithm designed for a multihop radio network should make the best use of the scarce system resource, the broadcast channel. That is, the main objective of any scheduling algorithm should be to obtain the shortest frame possible so as to maximise the network throughput. For deterministic tasks like broadcasting and gossiping this increased throughput lowers the delay in accomplishing the task. A short frame time can obviously be obtained through efficient time and spatial reuse of the channel. Also, the algorithm should be simple, fast and easy to implement. This means that it should have a lower rate of growth of running time, i.e., a good time complexity. In most cases the schedule is basically the result of a tradeoff between the two conflicting requirements, the performance (measured in terms of the length of the frame provided by the algorithm) and the time complexity of the scheduling algorithm. A good scheduling algorithm should try to provide a balance between the two giving the best tradeoff possible.

Chapter 4

Gossip Algorithms for Radio Networks

4.1 Introduction

In this chapter we develop some algorithms for gossiping in multihop packet radio networks and present various analyses results for these algorithms. We first consider arbitrarily connected radio networks, modelled as general graphs and give gossip algorithms for these graphs. Following the treatment of general graphs, we consider network topologies modelled by certain special classes of graphs. Specifically, we give the gossip schedule and time required to complete the schedule for complete graphs, star, ring and bus networks. Our choice of these special topologies from among many others is governed by their mention in the literature [31] and possible practical applications. In fact, it is claimed in [30] that star, bus and ring topologies are really some of the best choices available in the context of personal and indoor communication networks. For example, a mobile communication system for a highway can be modelled by a bus network [31, 55]. With the growing interest in the field of personal and mobile communication systems, the algorithms that we develop for these topologies would definitely be very useful for real networks.

Before proceeding further, we first give a precise statement of our problem and then explain the approach adopted in the development of the algorithms.

4.1.1 Problem Statement

We are given a multihop packet radio network with N stations connected in an arbitrary fashion. Each station has a single message that is to be communicated to all the other stations in the network. It is assumed that time on the network is slotted and the network is synchronised. The slot length is exactly equal to the message transmission time plus the maximum propagation delay between any two neighbours in the network and is the same for all nodes. All stations are synchronised to begin transmission in the beginning of a slot during which the communicating nodes can exchange exactly one packet consisting of a single message out of the several possible messages available in the station's buffer. The messages that may be contained in the station's buffers are those that have come from other stations in the network and need to be communicated to those stations that have not already received it. The problem is to develop gossip algorithms that minimise the time taken to gossip and delivers the gossip schedule in polynomial time.

4.1.2 Solution Approach

As was mentioned earlier, the primary objective of this dissertation is to develop polynomial time algorithms that generate schedules for gossiping in as few time slots as possible. In this quest we first need to examine the existence of polynomial time algorithms for gossiping in optimum time. Unfortunately the result available with us in this context is:

> For a radio network consisting of N stations, k to all broadcast is NP-hard for any $k \geq 1$ [45]

Gossiping is really a special case of k to all broadcast where $k = N$. Thus, the problem of finding an optimal gossip algorithm for radio networks is a NP-hard problem.

In such situations, a common approach is to look for good approximate solutions for the problem. In this thesis we propose some good approximation algorithms for gossiping in packet radio networks.

Since we have to provide approximate gossip algorithms, we would like to measure the quality of the approximation. One way to do it could be to analyse the "time complexity", and the "inpractice performance" of the algorithms. By time complexity, we mean the rate of growth of running time of the algorithms with increase in some parameters of

the network. We usually consider one algorithm to be more efficient than the other if its running time has a lower order of growth. In looking for lower order of growth, we should not forget that it should always be supplemented by a good inpractice performance and we should try to achieve an optimum balance between the two. For the algorithms that we develop in this thesis, we evaluate the inpractice performance by using realistic random inputs, that is by generating random graphs representing realistic radio networks according to the model discussed in Section 4.2. We then run the algorithms on these graphs to get the complete transmission schedule for all the stations in the network. To find the time taken to gossip using the schedule obtained as above, we have developed a simulation program. This program operates on a large number of random graphs for each gossip algorithm to estimate its inpractice performance.

In the following we discuss the approach that we have used in the development of our algorithms.

As must be clear from the discussion in the preceding chapters, the main issue in radio networks is to efficiently accomplish the task on a broadcast channel on which collisions may occur due to simultaneous transmissions. Depending on the needs of the communication requirement, we may or may not allow certain of these collisions. In all of our gossip algorithms we achieve the objective of gossiping by dividing the radio network into several areas. This division is done in such a manner that the needs of transmissions by stations in a given area are not affected by the needs of the transmissions by stations in other areas. To obtain a good time and spatial reuse of the channel some of our algorithms permit certain collisions in the network. It is to be noted here that we are also concerned about the complexity of the scheduling algorithm. and so we reduce it by limiting the number of nodes and links to be considered in the division process.

All the algorithms developed by us solve the problem of gossiping in a combination of frames called a gossip frame. The algorithms take an arbitrary connected graph $G = (V, E)$ as their input and using the rules of the algorithm assign a colour to each of the nodes present in a subset S of V: S depends on the gossip algorithm being used. This process of colouring gives us a "primary frame" which basically consists of a number of time slots, each slot corresponding to a colour. Repetition of this frame in time with further improvements during the repetition gives us the gossip frame that can achieve

gossip in the network. All the nodes possessing a particular colour, say c, will transmit during the slot corresponding to this colour c in the gossip frame. In addition to solving for a node the problem of "when to transmit" the algorithm also gives directions on "what to transmit".

4.2 Topology Modelling of Radio Networks

The study of any network algorithm often involves simulation or analysis using an abstraction or model of the actual network structure. The reason is quite obvious: networks that are large enough to be interesting are also expensive and difficult to control. Therefore, they are rarely available for experimental purposes. Moreover, it is generally more efficient to assess solutions using analysis or simulation – provided the model is a "good" abstraction of the real network. That is why the studies based on randomly-generated network models are very popular in practice. Our work also follows this approach to carry out a performance evaluation of the gossip algorithms that we have developed.

This section facilitates the description of our experimental results provided later in the chapter by giving the reader some requisite background. Here, we discuss the way we have modelled a packet radio network for experimentally studying the inpractice performance of our algorithms. The results provided are really the outputs obtained by giving some randomly generated graphs as inputs to the algorithms. However, in order to get a real picture of the inpractice performance of these algorithms, it is necessary that the graphs represent as 'realistic' a topology as possible. This necessitates the need for developing a good technique for generating the random graphs in the context of radio networks so that the results obtained should be as close to those in a real network as possible.

A rather widely used approach to generate random graphs is to use a simple probabilistic model. In this approach, a graph of a given number of vertices is generated by placing an edge between any two vertices according to some probability distribution. One way to go about this is to generate each possible edge out of a total of $N(N-1)$ edges for a graph having N vertices, with a fixed probability p. That is, it is assumed that each vertex is equally likely to be connected to all of the other vertices. This approach

has been used for example in [9] for evaluating the performance of some algorithms developed to solve the well known problem of vertex colouring. However in the case of radio networks where each station corresponds to a vertex, this is rather an unreasonable assumption. Here the connectivity between any two stations really depends on the geographical location of the stations i.e., more distant stations are likely to have a low value of connection probability. In practical networks each station has a certain transmission radius and all stations within this transmission range are suitable candidates to be connected to this station, or to be more precise, to be called its neighbours That is why any realistic model developed for a radio network should definitely take into account this unique property of "locality" of links.

Our experimental results obey the assumptions of a noiseless, immobile radio network in which all stations have the same transmission radius. In this context we may represent any radio network by the 3-tuple (N, R, P) where N is the number of stations, R is the transmission range of each station and $P = \{(x_i, y_i) : \quad 1 \leq i \leq N\}$ is the set of geographical locations for each of the stations. This set is generated randomly by using a uniform distribution for the x and y coordinates of each of the stations. After having generated the random network, the graph, $G = (V, E)$, corresponding to this network may be obtained by applying the following rule to obtain the edge set E.

> Any edge (u, v) may be considered as belonging to edge set E if and only if the Euclidean distance between (x_u, y_u) and (x_v, y_v) is less than or equal to R.

The node set V corresponds to the stations in the network.

The above rule gives us an undirected graph which can be used to study any algorithm for radio networks. Arguably, this model looks sufficiently realistic for an accurate performance evaluation of the various gossip algorithms.

4.3 Collision Free Gossip Algorithm

Let us assume that the given radio network is represented by the graph $G = (V, E)$. Let us assume that $\mid V \mid = N$ and ρ denote the maximum degree present in the graph. Then the fundamental idea behind this algorithm is to develop the primary frame required by

distance-2 colouring of the graph $G = (V, E)$. The desired gossip frame for gossiping will be basically a repetition of this primary frame with some of the colours, we call these the "transient colours", making their presence only once in the entire gossip frame.

4.3.1 Algorithm Description

We first describe the procedure, Dis_2_colour, used by us for distance-2 colouring of the graph G. This colouring helps us in obtaining the primary frame for the algorithm and also to designate the transient colours. The main idea in this procedure is to colour the vertices in a certain order, based on the vertex degrees In particular, we first assign a unique label to each vertex in V. The labelling is performed in a progressive fashion. Suppose a vertex v is labelled l. The next vertex chosen will be the one having the least total number of neighbours in the subgraph formed by the unlabelled vertices and will be labelled $l + 1$. After labelling all of the vertices in this manner, we colour the vertices by considering them in decreasing order of their labels. We assign the lowest numbered colour that can be assigned without causing a conflict with previously coloured vertices. For a particular vertex, all vertices having a distance less than or equal to two are said to be conflicting with it. That is why the colouring is called distance-2 colouring. By colouring in this manner the guarantee that the algorithm can provide is that the transmission of a particular node in its assigned slot (depending upon its colour) would definitely reach collision free to all its neighbours. This algorithm can thus be seen to be characterized as a "minimum degree last" algorithm. Let us now derive the procedure for identifying the transient colours in the primary frame, which as previously stated should appear only once in the gossip frame.

For finding out whether a given colour is transient or not, the algorithm requires to investigate the degrees of the nodes possessing this colour. If all the degrees investigated have a value equal to 1, signifying that all the nodes possessing this colour have a single neighbour, then the colour will be transient. The motivation behind this concept of transient colours is that if a node has a single neighbour in the network, then it should obviously be allowed to transmit only once in the whole process, where it will transmit its own message. There is no need to have this node transmit again because in subsequent transmissions it will only be going to transmit the messages that are already available

```
Procedure label(G)
/*
The input to this procedure will be the undirected graph G = (V, E) and the
output will be an assignment of label to each of the vertex in the graph.
*/
    begin
        set count ← 1
        while (there is any vertex which is not labelled) do
            u ← minimum degree vertex in the subgraph
                induced by unlabelled vertices
            label(u) ← count
            count ← count + 1
        endwhile
    end
```

Figure 4.1: Pseudocode for procedure label

with its neighbours.

Summarising, the tasks to be performed by Dis_2_colour procedure are to firstly label the vertices, (using the procedure label), assign colours based on these labels and identify the transient colours.

It can be easily seen from Figure 4.1 that the procedure label terminates and that each vertex is labelled exactly once. Also, since the count is incremented in every step, each vertex is assigned a unique label.

Clearly, the largest colour that is assigned to a vertex u is at most one greater then the maximum size of the set Y, constructed in the Dis_2_colour procedure in each iteration (see Figure 4.2). For a graph with maximum degree ρ, there can be at most ρ adjacent vertices and $(\rho(\rho-1)$ distance-2 vertices. Assuming in the worst case that all of these vertices are coloured differently, number of colours used in the worst case is no more than $(1+\rho)+\rho(\rho-1) = \rho^2+1$. An improved bound is provided later in this section.

Running the Dis_2_colour procedure on the graph G gives us a primary frame which can be repeated in time to give the gossip frame. The colours corresponding to the slots in the gossip frame provides an answer to the question: Which nodes can transmit in each of the slots of the gossip frame ?. The question that still needs to be answered is: What will the nodes transmit in these slots ? This question can be answered by providing a rule according to which the nodes may select one message out of the several messages they may possibly have at the beginning of the slot. The following are some of the options:

- *First Come First Serve (FCFS)* where the node selects a feasible message which arrived at it no later than any other feasible message.
- *Last Come First Serve(LCFS)* where the node selects a feasible message that arrived at it no earlier than any other feasible message.
- *In Order(IO)* where the node chooses the feasible message with the smallest identity (source node) number.
- *Random Order of Service(ROS)* where each feasible message has an equal probability of being selected.

The rules can either be strictly followed or the nodes may give first priority to their own messages and then adhere to these rules. This gives rise to a total of seven selection methods that the nodes may employ for selecting a message out of several messages in the buffer they may have. Let us call these rules by the names FCFS, LCFS, IOS, ROS, LCFS(s), IOS(s) and ROS(s), the 's' in the bracket signifying that the rules are to be followed strictly. Whatever rule of selection be adopted, the guarantee the algorithm provides is that no message need to be transmitted more than once by a node. In the worst case each node needs to transmit at most N times before gossip is complete. However the actual number of transmissions will be sufficiently smaller than that.

Having the entire information about the gossip schedule the schedule can then be run at the centralised supervisor to get an exact number of slots required for gossiping.

To explain the operation of this algorithm more clearly let us apply it to a sample graph shown in Fig 4.3(a). Figures 4.3(b)-(i) illustrate the operation of the procedure

Procedure Dis_2_colour(G)

/*

The procedure takes an arbitrary connected undirected graph $G = (V, E)$ as its input and firstly calls the procedure label to label the vertices of G. The output will be a colouring scheme $c : V \rightarrow \{1, 2, 3 \ ..\}$ and a flag $f(c)$ for each colour c. If $f(c) = 0$, the colour c is transient

*/

```
begin
    label(G)
    for (each u ∈ V) set c(u) ← 0. endfor
    for (every colour c which can be assigned) set f(c) ← 0. endfor
    for (j = N to 1)
        pick the vertex u with label j
        set Y ← φ
        for (every v which is a 1-hop or 2-hop neighbour of u)
            Y ← Y ∪ {c(v)}
        endfor
        c(u) ← (least natural number y such that y ∉ Y)
        if (degree(u) > 1) set f(c(u)) ← 1 endif
    endfor
end
```

Figure 4.2: Pseudocode for procedure Dis_2_colour

by deleting all the nodes which have been labelled in the previous iterations, i.e., iterations upto $j-1$.

We now illustrate the procedure Dis_2_colour for assigning the colours to the nodes in the graph. Initially the algorithm assigns to all nodes present in the graph, colour 0. Further iterations involved in the process are as shown in Figure 4.4. In the discussion we assume that a coloured node is one which has been assigned a colour other than 0 and the assignable colours are those from the set $C = \{1, 2, 3 \ldots, \}$. The shaded nodes in Figures 4.4(b)-(i) signify the nodes conflicting with the node being coloured in the iteration.

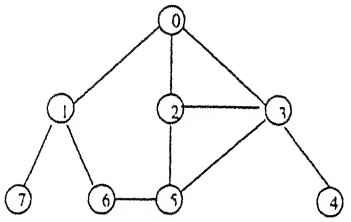
It is easily seen that none of the six colours is transient because the only degree-1 nodes in the graph are nodes 4 and 7 which are assigned colours 5 and 1 respectively. However some higher degree nodes also possess these colours, which forces us not to declare these colours as transient. Thus the primary frame for this graph consists of six time slots.

4.3.2 Properties of the Algorithm

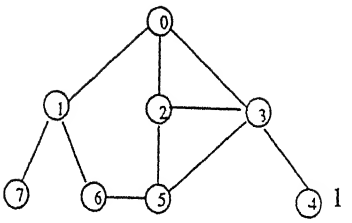
We first prove that the procedure Dis_2_colour can be executed in polynomial time and then provide bounds on the number of slots in the primary frame and the gossip frame.

Property 1 *The execution time complexity of procedure* Dis_2_colour *is* $O(N^2 + N\rho^2)$.

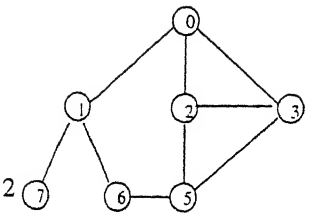
Proof: The running time of the procedure Dis_2_colour may be evaluated in two parts: the running time of the procedure label and the running time of the remainder of the algorithm. In procedure label, finding a vertex of minimum degree takes $O(N)$ time and updating the degrees of the neighbours on deletion takes $O(\rho)$ time, using a straightforward implementation. Since the above two operations are performed for every vertex, it follows that the running time of the procedure label is $O(N(N+\rho))$. To evaluate the complexity of the remainder of the algorithm, we first note that the process of colouring a node v requires to visit all its neighbours having a maximum value of ρ and the neighbours of these neighbours which can at most be equal to $\rho(\rho-1)$, as one of them is the node v itself, giving a maximum number of $O(\rho^2)$ visits for any vertex



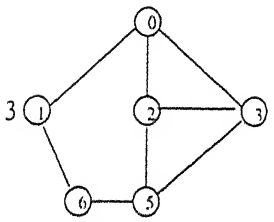
(a) Initially



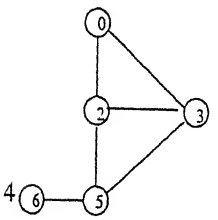
(b) Iteration 1



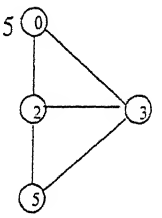
(c) Iteration 2



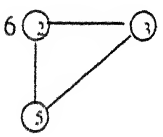
(d) Iteration 3



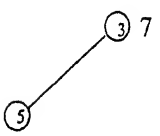
(e) Iteration 4



(f) Iteration 5



(g) Iteration 6



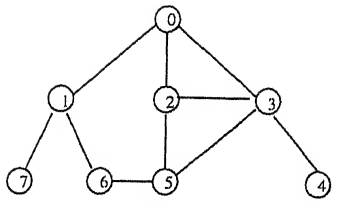
(h) Iteration 7

(i) Iteration 8

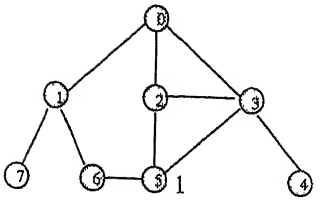
1. Search for minimum degree node. Find nodes 4 and 7 out of which arbitrarily choose 4. Assign label 1 to node 4. Remove node 4 and generate subgraph S_2 of (c).
2. Choose node 7 and label it 2. Generate subgraph S_3 of (d).
3. In S_3 nodes 1 and 6 have a minimum degree of 2. Arbitrarily choose 1 and assign it label 3. Generate subgraph S_4 of (e).
4. Choose node 6 and label it 4. Generate subgraph S_5 of (f).
5. Again, nodes 0 and 5 in S_5 have the minimum degree. Choose 0; assign it label 5 and generate S_6 of (g).

In the next three steps assign labels 6, 7 and 8 to nodes 2, 3 and 5 respectively. Since all nodes have been labelled this brings about an end to the procedure 'label'.

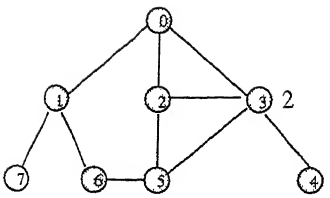
Figure 4.3: Illustrating the operations of the procedure label on a sample graph



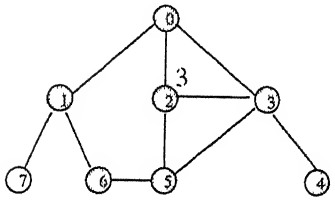
(a) Initially



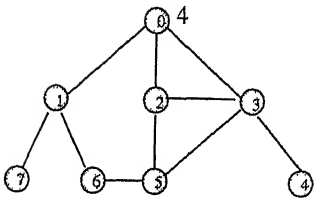
(b) Iteration 1



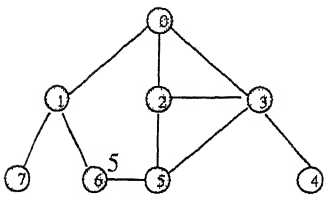
(c) Iteration 2



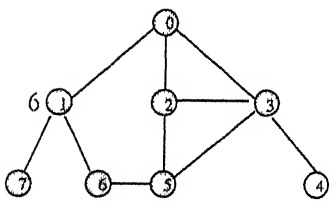
(d) Iteration 3



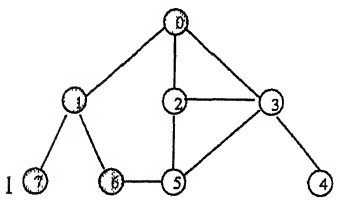
(e) Iteration 4



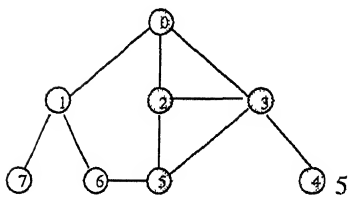
(f) Iteration 5



(g) Iteration 6



(h) Iteration 7



(i) Iteration 8

1. Node 5 with the maximum label of 8 is chosen for colouring and since none of its conflicting nodes have been assigned a colour it is assigned colour 1.
2. Node 3 is chosen. Its conflicting nodes are nodes 0, 1, 2, 4, 5 and 6. Since 5 has been assigned colour 1, it is assigned colour 2.
3. Node 2 with nodes 0, 1, 3, 4, 5 and 6 as its conflicting nodes is chosen for colouring and assigned colour 3.
4. Node 0 is chosen next and since it is within distance 2 from all the nodes in the network, it is assigned colour 4.
5. Node 6 is chosen for colouring and is assigned the least available non conflicting colour 5.

In a similar manner in the next three iterations, nodes 1, 7 and 4 are assigned colours 6, 1 and 5 respectively.

Figure 4.4: Illustrating the operations of the procedure Dis_2_colour on the sample graph

v. The least entry y from the set Y containing the conflicting colours of the vertex v can again be found in $O(\rho^2+1)$ time by keeping an entry corresponding to each of the possible available colour which can obviously be no more than (ρ^2+1). Thus the whole process takes $O(\rho^2)$ time steps. Since this is iterated for every vertex, it follows that the complexity of the for loop in which the nodes are coloured is $O(N\rho^2)$ in the worst case. Thus the overall time complexity, with only the simplest of data structures is $O(N^2+N\rho+N\rho^2)$ or $O(N^2+N\rho^2)$. □

Before providing a bound on the length of the primary frame, we prove the following lemma

Lemma 1 *In the labelling of an arbitrary graph having thickness t, every vertex v can have at most $6t-1$ neighbours labelled larger than v*

Proof: In each iteration i of the procedure label we look for a vertex v having the least possible degree in the subgraph S_i. This subgraph S_i consists of all unlabelled vertices left up till iteration i. Now, since every graph of thickness t contains at least one vertex of degree $6t-1$ or less [41], v, the minimum degree vertex can have at most $6t-1$ neighbours. Since, all these neighbours are to be labelled in subsequent iterations with labels larger than v's label, the lemma follows. □

Property 2 *The algorithm* Dis_2_colour *does not assign more than* $\rho(6t-1)+(6t-2)(\rho-6t+1)+1$ *colours to any graph G of thickness t.*

Proof: Let m be the label of a vertex u that we colour in iteration $N-m+1$ of the procedure Dis_2_colour. Let us partition the neighbours of u into two sets U_1 and U_2. The set U_1 consists of vertices labelled labels larger than m while set U_2 consists of the remaining vertices, labelled smaller than m. The proof is given by counting the number of colours precluded by nodes in the two sets.

If j be the cardinality of set U_1, then from Lemma 1, it follows that $j \leq 6t-1$. Thus in the worst case, the number of colours precluded for u by the vertices present in U_1 and their neighbours can be no greater than $j+(\rho-1)j = j\rho$.

Now let us calculate the number of colours conflicting due to $\rho-j$ vertices present in set U_2. From Lemma 1, each vertex in U_2 can have at most $6t-1$ vertices labelled larger than itself one of which is the vertex u itself. Hence, corresponding to each vertex in U_2,

we have to consider the colours of $6t-2$ of its neighbours. If each of these is labelled larger than u and has a distinct colour, then at most $(\rho-j)(6t-2)$ colours are to be excluded in colouring due to nodes present in the set U_2.

Thus in colouring u at most $\rho j+(\rho-j)(6t-2)$ colours have to be excluded. From Lemma 1 $j \leq 6t-1$, and hence the upper bound on the number of colours in the primary frame will be $\rho(6t-1)+(6t-2)(\rho-6t+1)+1$. □

Property 3 *The gossip frame length is upper bounded by* $ND(\rho(6t-1)+(6t-2)(\rho-6t+1)+1)$ *where* D *is the diameter of the graph* G.

Proof: This bound is rather loose and the proof is straightforward. The message from each node will travel to all vertices in the graph in at most D frames. If each node in the graph initiates its broadcast after the completion of a previous broadcast initiation, a total of ND primary frames would be required in the gossip frame. □.

4.3.3 Performance Results

We now present the results of simulation experiments using this algorithm. Table 4.1 shows the number of slots required to complete the gossiping problem for each of these seven selection methods. Each entry gives the average number obtained by running the algorithm over fifty randomly generated graphs.

4.4 Centralised Spanning Tree Algorithm

In contrast to the previous algorithm in which obeying the developed schedule guarantees the successful reception of a node's transmission at all its neighbours, this algorithm proceeds by assigning to each node in the network a particular node called it's parent. This parent is assigned the duty of transmitting any message in the network to its children. The algorithm also guarantees that if the children of a particular parent transmit in their respective transmission slot, then the parent would not encounter any collision in this slot. The algorithm constructs a spanning tree T and assigns colours to all the nodes in V in such a way that no node's transmission will collide in its parent or children in the tree. As in the Dis_2_colour algorithm, the set of assigned colours gives a primary frame

N	R	FCFS	LCFS	IOS	ROS	LCFS(s)	IOS(s)	ROS(s)
10	0.3	53	56	55	55	58	55	53
10	0.4	59	61	60	60	67	61	61
10	0.5	77	80	78	63	82	81	80
10	0.6	83	86	87	66	89	88	83
10	0.7	85	88	89	68	92	90	87
20	0.3	157	171	163	148	174	165	157
20	0.4	231	242	240	213	251	244	234
20	0.5	296	309	309	228	322	312	285
20	0.6	325	355	355	236	372	361	332
20	0.7	338	367	368	245	389	371	341
50	0.3	962	982	980	857	999	987	955
50	0.4	1418	1450	1452	898	1477	1461	1399
50	0.5	1902	1964	1963	982	2001	1973	1884
50	0.6	2145	2245	2244	1013	2287	2262	2173
50	0.7	2155	2349	2349	1037	2396	2367	2236
100	0.3	3685	3726	3730	2438	3763	3745	3650
100	0.4	5711	5807	5811	2513	5867	5821	5658
100	0.5	7695	7935	7928	2681	8009	7949	7687
100	0.6	9107	9374	9372	2734	9471	9402	9087
100	0.7	9182	9794	9796	2745	9889	9822	9494
200	0.3	14629	14680	14702	6273	14749	14721	14585
200	0.4	22770	23166	23176	6578	23280	23165	22844
200	0.5	31010	31778	31756	6732	31925	31783	31392
200	0.6	36043	37154	37145	6894	37348	37211	36556
200	0.7	37171	38782	38781	6907	38983	38793	38074

Table 4.1: Simulation results depicting the inpractice performance of the "Collision Free Gossip Algorithm"

which is repeated in time according to certain rules described later to yield the gossip frame.

4.4.1 Algorithm Description

This algorithm uses a procedure Cent_Tree to assign colours to the nodes, identify the transient colours and also establish the parent-child relationship in the input graph $G = (V, E)$. The procedure Cent_Tree starts with an arbitrary node in the graph called the root node. Any node in the network could be chosen as the root node without affecting the correctness. The procedure constructs a spanning tree, T, on this graph and assign the colours to the nodes as it progressively constructs T from the input graph G. Associated with each node in the graph is a data structure that contains the colour of the node and its parent.

The procedure Cent_Tree begins with the spanning tree T being empty. In the first iteration the root node r will be picked up and assigned colour 1. It will then be added into the tree with the subsequent adoption of all its neighbours as its children. This is done by making the entry p corresponding to the parent in the data structure of all its neighbours as r. In subsequent iterations the algorithm repeatedly chooses an uncoloured node v, assigns it a colour according to the colouring rules described below, adds it to the spanning tree T and finally assigns v to be the parent of all its orphan neighbours, neighbours that have not yet been assigned a parent. The algorithm stops when all the nodes have been assigned a colour. For each iteration, the node selected to be coloured should be the first uncolured node to have a parent in terms of the iteration number in which the assignment was done. Since, the algorithm requires each node to be coloured the whole process can be completed in N iterations - N denoting the number of stations in the radio network.

A node v is coloured with the lowest natural number, c, that satisfies the following three conditions:

c1a: c is not the colour of v's parent $p(v)$.

c1b: c is not the colour of any of neighbour of $p(v)$.

c1c: c is not the colour of any of v's neighbour parent.

The first two conditions basically ensure that each parent will be successfully able to get its children transmissions while the third one ensures that every transmission of a parent is received collision free by its children.

A colour c is transient if all the nodes having this colour have no children. These colours appear only once in the gossip frame in which the nodes possessing these colours transmit their own message. The gossip frame is constructed in the same manner as in the collision free gossip algorithm. This answers the question for every node in the network "when to transmit". As in the previous algorithm, to answer the question of "what to transmit" any of the techniques i.e., FCFS, LCFS, IOS, ROS, LCFS(s), IOS(s) and ROS(s) described previously can be used. The exact length of the gossip frame can be obtained by running the gossip schedule at the centralised supervisor.

The pseudocode for the procedure Cent_Tree is given below. In this description, an uncoloured node is one that is coloured 0. $f(c)$ for any colour c indicates whether the colour is transient.

The Cent_Tree procedure is applied to the sample graph and the colouring mechanism is illustrated in Figure 4.6. Node 0 is chosen as the root node. The shaded nodes in each figure represent the nodes conflicting to the node chosen for colouring in the particular iteration.

4.4.2 Properties of the Algorithm

Property 4 *The execution time complexity of procedure* Cent_Tree *is* $O(\rho N)$.

Proof: For a node v to be coloured in the graph, the procedure Cent_Tree requires to visit its parent $p(v)$, $p(v)$'s neighbours, which can be at most $(\rho - 1)$, and the parents of all of v's neighbours which can again be at most $(\rho - 1)$ since $p(v)$ is also a neighbour of v. Thus a maximum of $(2\rho - 1)$ visits will be made for colouring any vertex $v \in V$. If ρ is the maximum degree in the graph, from Property 5, the number of distinct colours used is upper bounded by 2ρ. Thus the least available non conflicting colour can be found in at most $O(2\rho)$ time steps, by keeping a flag for each assignable colour to v. Since the algorithm requires each node to be coloured, in the worst case the overall colouring mechanism would have a time complexity equal to $O(N\rho)$. The algorithm also requires to visit all the nodes to evaluate $f(c)$ for each assignable colour c. This procedure obviously

Procedure Cent_Tree(G)

/*

The input to this procedure is an arbitrarily connected undirected graph $G = (V, E)$

The output is an assignment of colours $c : V \rightarrow S$, $S = \{1, 2, 3, \ldots\}$

Each colour has a flag, $f(c)$, to indicate if it is transient.

If $f(c) = 0$, colour c is transient

*/

begin

for (all $v \in V$) set $c(v) \leftarrow 0$ endfor

for (every colour c that can be assigned) set $f(c) \leftarrow 0$ endfor

select a root node r

set $T \leftarrow \phi$

set $c(r) \leftarrow 1$; set $f(1) \leftarrow 1$; $T \leftarrow T \bigcup \{r\}$

for (each neighbour, v, of r) set $p(v) \leftarrow r$ endfor

while (there is an uncoloured vertex in V) do

select v, the first uncoloured node that has been assigned a parent

$T \leftarrow T \bigcup \{v\}$

select the lowest natural number i such that

c.1: i has not been assigned to v's parent

c.2: i has not been assigned to a neighbour of v's parent

c.3: i has not been assigned to a parent of v's neighbour

set $c(v) \leftarrow i$.

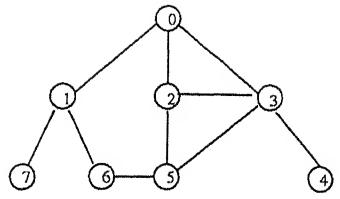
for (each neighbour, u, of v which does not have a parent), set $p(u) \leftarrow v$ endfor

if (at least one such neighbour is obtained) set $f(i) \leftarrow 1$ endif

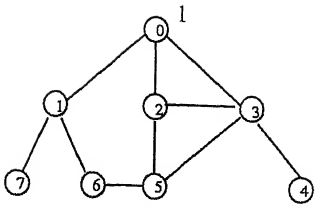
endwhile

end

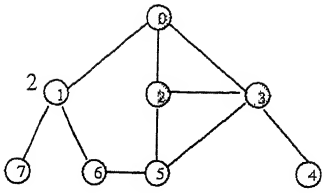
Figure 4.5: Pseudocode for Cent_Tree procedure



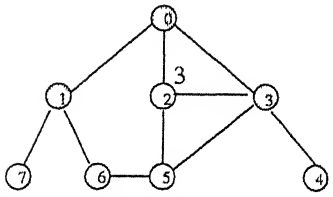
(a) Initially



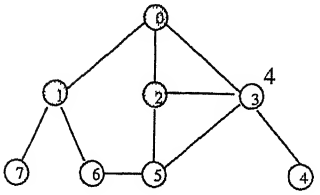
(b) Iteration 1



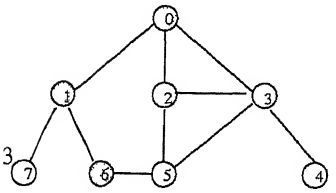
(c) Iteration 2



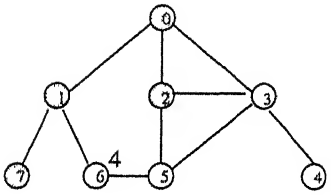
(d) Iteration 3



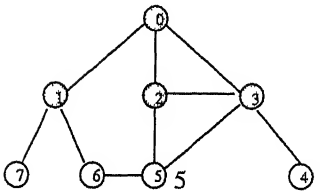
(e) Iteration 4



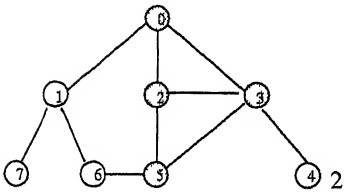
(f) Iteration 5



(g) Iteration 6



(h) Iteration 7



(i) Iteration 8

1. Root i.e., node 0 is assigned colour 1, added into the tree and all its neighbours i.e. nodes 1, 2 and 3 are assigned 0 as their parent.

2. Of the possible choices of 1, 2 and 3, arbitrarily choose node 1. **c1a** precludes colour 1 and so colour 2 is assigned to it. It is assigned to be the parent of nodes 6 and 7. $T = T \bigcup \{1\}$.

3. Among 2 or 3, choose node 2. **c1a** and **c1c** preclude colour 1; **c1b** precludes colour 2 and so it is coloured 3. Node 5 will be its child. $T = T \bigcup \{2\}$.

4. Pick node 3. **c1a** and **c1c** preclude colour 1; **c1b** precludes colour 2; **c1b** and **c1c** precludes colour 3. So, colour 4 is assigned to it and node 4 is assigned to be its child. $T = T \bigcup \{3\}$.

5. From nodes 6 and 7, pick node 7 for colouring. **c1a** precludes colour 2; **c1b** and **c1c** disallow colour 1. So, colour 3 is assigned to it. It is not assigned any children. $T = T \bigcup \{7\}$.

In a similar manner, in the next three iterations, nodes 6, 5 and 4 are assigned colours 4, 5 and 2 respectively and added into T. It can be easily seen that colour 5 comes out to be the transient colour.

Figure 4.6: Illustrating the operations of the procedure Cent_Tree on the sample graph

has a time complexity of $O(N)$. Thus the overall worst case time complexity of the procedure Cent_Tree is $O(\rho N)$. □

Property 5 *For any graph $G = (V, E)$ with a maximum degree ρ the total number of distinct colours assigned by the procedure* Cent_Tree *is upper bounded by 2ρ.*

Proof: To prove this property we need to show that for any node in the graph $G = (V, E)$ there always exist a colour with a sequential number lower than $2\rho + 1$ which meets the conditions required to be assignable to it.

Let v be the node to be coloured in a particular iteration. Let $p(v)$ be the parent of v. Let $S_1(v)$ denote the set of the neighbours of $p(v)$ and let $S_2(v)$ denote the set of the parents of v's neighbours. Both the sets can have a maximum cardinality of ρ. As is quite obvious, one of the entries in the set $S_1(v)$ will be v itself. Also, $p(p(v))$ will be a node common to both $S_1(v)$ and $S_2(v)$. In the worst case, assuming that all the nodes of $S_1(v)$ and $S_2(v)$ are coloured differently the colouring rules **c1b** and **c1c** preclude $(2\rho - 2)$ colours for v. Rule **c1a** precludes the parent's colour too. Thus v can be assigned the remaining $(2\rho)^{\text{th}}$ colour. □

Property 6 *The gossip frame length in the centralised spanning tree algorithm is upper bounded by $2ND\rho$, where D is the diameter of the graph.*

Proof: The proof is along the same lines as that of Property 3. □

4.4.3 Performance Results

We now present the results of simulation experiments using this algorithm. Table 4.2 shows the number of slots required to complete the gossiping problem for each of the seven selection methods. Each entry gives the average number obtained by running the algorithm over fifty randomly generated graphs.

N	R	FCFS	LCFS	IOS	ROS	LCFS(s)	IOS(s)	ROS(s)
10	0.3	41	46	43	42	48	45	43
10	0.4	34	37	37	35	39	37	36
10	0.5	32	32	33	32	34	33	32
10	0.6	26	26	26	26	27	26	26
10	0.7	25	25	25	25	26	25	25
20	0.3	113	121	116	115	123	117	115
20	0.4	109	113	111	109	116	112	109
20	0.5	95	95	96	95	99	96	95
20	0.6	68	68	68	68	69	68	67
20	0.7	52	51	52	52	52	52	52
50	0.3	491	501	497	485	506	498	487
50	0.4	396	397	398	393	403	398	395
50	0.5	286	287	287	286	291	287	286
50	0.6	235	235	235	233	238	235	235
50	0.7	121	121	121	121	121	121	120
100	0.3	1448	1462	1458	1371	1470	1458	1428
100	0.4	987	990	992	970	999	992	974
100	0.5	722	722	723	718	728	723	719
100	0.6	564	564	564	560	567	564	560
100	0.7	256	256	256	256	257	256	256
200	0.3	4202	4215	4218	3831	4235	4218	4096
200	0.4	2541	2547	2549	2491	2559	2549	2523
200	0.5	1661	1662	1662	1633	1669	1662	1647
200	0.6	1096	1096	1096	1095	1099	1096	1096
200	0.7	656	656	656	656	657	656	656

Table 4.2: Simulation results depicting the inpractice performance of the "Centralised Spanning Tree Gossip Algorithm"

4.5 Gather-Scatter Algorithm

This algorithm achieves gossiping by first gathering the messages from all the nodes in the network at a root node through an accumulation algorithm followed by the scattering of these messages to all the nodes in the network through a dissemination algorithm. The root node can be chosen arbitrarily without affecting the correctness of the algorithm but the performance would really depend upon the choice. More specifically, the algorithm would yield the best performance if we choose the central node in the graph as the root node. All our experimental results concerning the algorithm follow this particular choice.

4.5.1 Accumulation Algorithm

This algorithm collects all the messages from the network nodes at a particular node called the root node. Without loss of generality let us choose the central node in the graph $G = (V, E)$ representing the radio network to be the root node. We first assign to each node in the graph (excluding the root node) a colour c and a number f. Assignment of colours to the nodes allows us to construct a primary frame consisting of a number of time slots each slot corresponding to an assigned colour. Let us denote this primary frame by the set $L = \{1, 2, \ldots, l\}$, where each colour c present in the frame is the colour assigned to some subset $S(c)$ of V. Each colour present in the frame is also assigned a counter, $C(c)$. The counter $C(c)$ for a particular colour c depends on the values of f assigned to the nodes of the set $S(c)$. (The way the assignment of f to the nodes is done will be discussed later.) The counter assignment procedure will again give rise to a set, $I = \{C(1), C(2), \ldots, C(l)\}$. Based on these two sets L and I, an accumulation frame is built as follows. Repeat the primary frame L with each colour c making its appearance in the accumulation frame exactly $C(c)$ times. Based on the above approach, it can be seen that length of the accumulation frame is exactly

$$\sum_{i=1}^{l} C(i)$$

number of time slots, l denoting the cardinality of the set L. This bound can be achieved by having the transmissions strictly adhere to the following rule:

> **Relay Rule:** In each timeslot c, all nodes scheduled to transmit in c, should in the first chance transmit its message and then should transmit only those

messages that have come from its children. Each node should transmit a message only once in the accumulation frame.

We now give a detailed description of this accumulation algorithm. As can be seen the basic tasks performed by this algorithm are:

t1 Assignment of colours to the nodes to construct the required primary frame. This is performed by the procedure Acc_colour.

t2 Evaluate the counter corresponding to each colour of the primary frame. This is performed by the procedure Counter_assignment.

The procedure Acc_colour constructs a spanning tree T directed from the root node. In the process, the procedure establishes the parent-child relationship between the nodes. It also assigns colours to the nodes in a manner that guarantees the reception by the parents of the messages transmitted by the children if the children transmit in their respective transmission slots. In other words, the algorithm guarantees that when a node transmits in its transmission slot (decided by its colour) its parent will successfully receive the transmission. Note that the algorithm does not care whether nodes other than the parents receive the transmission in the slot. The root node is not obviously assigned any colour.

The procedure Acc_colour starts with the tree T being empty. In iteration 1 the root node r is picked up and added into the tree. This root node now becomes the parent of all its neighbours in G and these neighbours are assigned r as their parent. One of these neighbours, say u, is selected in the next iteration and assigned a colour (based on the colouring rules below). All the orphan neighbours of u will be assigned u as their parent, and u will be added to the tree. In subsequent iterations, in the beginning there will be a set of nodes U in V which have not been added into T. The procedure Acc_colour repeatedly chooses a node $u \in U$, assigns it a colour according to the colouring rules, makes u the parent of all the orphan neighbours of u, and adds it to T. In each iteration, the choice u is determined on the basis of the time it was assigned a parent. More specifically, nodes that are assigned a parent earliest are chosen for colouring.

The lowest natural number, c, satisfying the following rules is to be used when colouring a node.

c2a The colour c has not been assigned to the parent $p(v)$ of v.

c2b The colour c has not been assigned to any of neighbours of $p(v)$.

These colouring rules ensure that every parent successfully receives the transmission of its children, if they transmit in their scheduled time slots.

The procedure Counter_assignment is responsible for giving directions on how to iterate the primary frame created by the procedure Acc_colour in time to obtain the accumulation frame. The algorithm accomplishes this by assigning to each colour c of the primary frame a counter $C(c)$. Then the accumulation frame will be a repetition of the primary frame with each colour c appearing in the accumulation frame exactly $C(c)$ times. To be more specific, the basic structure and size of the primary frame will vary in each repetition with some colours making their presence while others being absent.

We now describe how to obtain the counters corresponding to each of the colours. For each node $v \in V$, let us call $f(v)$ as the number of "future generation nodes" in its family consisting of v, its children, their children and so on. The procedure Counter_assignment finds the entry $f(v)$ for all $v \in V$. This can be done by first chosing a node say $v \in V$. Now to calculate the entry $f(v)$ for v, the procedure visits all its children adds up the entries $f(v)$ corresponding to each of its children and further adds one corresponding to itself. By repeating this procedure of choosing a node, visiting its children and adding up the corresponding f's, $f(v)$ for all $v \in V$ can be obtained. The key to the correctness of the procedure is the process of chosing of nodes for the assignment of $f(v)$. The procedure Counter_assignment requires that the process of selecting the nodes to assign the entry f should be strictly in the reverse of the order in which they were assigned a parent in the Acc_colour procedure which is also the order in which they were entered into the tree T. So if we assume that the nodes were assigned parents in the order $O_1, O_2, \ldots, O_{N-1}$ then they should be picked in the order $O_{N-1}, \ldots, O_2, O_1$ for assignment of $f(v)$.

After the above assignment the value of $C(c)$ for a colour c is obtained as the maximum of the $f(v)$'s of all the nodes $v \in S(c)$.

It is easy to see that each node v has to transmit exactly $f(v)$ messages to complete the accumulation process. By adhering to the **Relay rule** each node is guaranteed to get a new message to transmit after every primary frame which it sends on the path towards the root node in a subsequent frame. This means that each node gets a chance

Procedure Acc_colour(G)

/*

The procedure takes an arbitrary connected undirected graph $G = (V, E)$ as its input.
The output is a colour assignment to the nodes of G and an array O where
the entry $O[i]$ represents the node coloured in iteration $(i + 1)$.

*/

begin

for(all $v \in V$) set $c(v) \leftarrow 0$. endfor

set $T \leftarrow \phi$; set *count* $\leftarrow 1$

$T \leftarrow T \cup \{r\}$

assign r as the parent of all its neighbours

while (there is an uncoloured vertex in $V - \{r\}$) do

select u, the first node in $V - \{r\}$ to be assigned a parent

$T \leftarrow T \cup \{u\}$

$O[count] \leftarrow u$

colour u with the lowest natural number i satisfying the following two conditions

c2a: i has not been assigned to u's parent

c2b: i has not been assigned to a neighbour of u's parent

for all of u's orphan neighbours assign u as the parent

count $\leftarrow$ *count* $+ 1$

endwhile

end

Figure 4.7: Pseudocode for Acc_colour procedure

to transmit $f(v)$ times in an accumulation frame and this chance can come in consecutive frames. We can now determine the number of slots used in the accumulation frame to be exactly equal to

$$\sum_{i=1}^{l} C(i)$$

where l is the number of colours assigned by the Acc_colour procedure.

The operation of the algorithm is illustrated on the sample network in Figure 4.9. Fig 4.9(a) shows the sample network and Figures 4.9(b)-(i) illustrate the iterations of the algorithm. The evolution of the array O is also shown. The shaded nodes in the figure represent the ones conflicting to the node selected in the iteration. Node 0 is chosen as the root node.

At the end of the process it can be seen that all the nodes have been coloured and the only colours used are 1, 2, 3 and 4. This essentially leads to the construction of a primary frame consisting of 4 time slots corresponding to the 4 colours assigned to the nodes. After the assignment of colours to the nodes in the network by the Acc_colour procedure a counter needs to be assigned by the Counter_assignment procedure to each of the four colours in the primary frame. By picking up the nodes in the reverse order in which they were entered into T, and then calculating the entry $f(v)$ for each of them, also evaluating the counters in the process, it can be verified that the procedure Counter_assignment assigns $C(1) = 3$, $C(2) = 2$, $C(3) = 2$ and $C(4) = 1$. This leads to a total of 8 slots required for accumulating all the messages in the root node 0.

4.5.2 Properties of the Algorithm

Property 7 *The execution time complexity of procedures* Acc_colour *and* Counter_assignmen *is* $O(\rho N)$.

Proof: Recall that the procedure used to obtain the accumulation frame is to first colour the vertices using Acc_colour procedure to obtain the primary frame and then obtain the number of times a given colour is to be repeated in the accumulation frame. This means that to evaluate the running time of the accumulation algorithm we have to evaluate the running times of the Acc_colour procedure and the Counter_assignment procedure seperately.

Procedure Counter_Assignment(G)

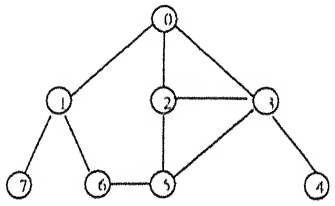
/*

The procedure takes an arbitrary connected coloured graph $G = (V, E)$ and
the array O constructed in the Acc_colour procedure as its input
The output is an assignment of counter to each of the colours assigned to nodes in V.

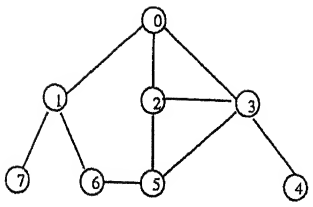
*/

```
begin
    for (all v ∈ V − {r}) set f(v) ← 1. endfor
    for (each colour c assigned to nodes in V) set C(c) ← 0 endfor
    set count ← N − 1
    while (count > 0) do
        set u ← O[count]
        for (each neighbour n, a child of u) f(u) ← f(u) + f(n) endfor
        let c be the colour of u
        if (C(c) < f(u)) then
            set C(c) ← f(u) endif
        count ← count − 1
    endwhile
end
```

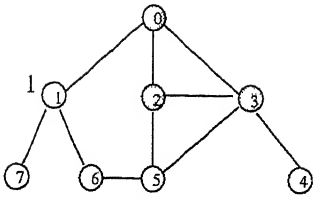
Figure 4.8: Pseudocode for Counter_assignment procedure



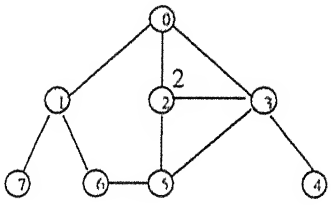
(a) Initially



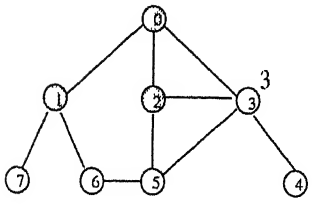
(b) Iteration 1 O={1,2,3}



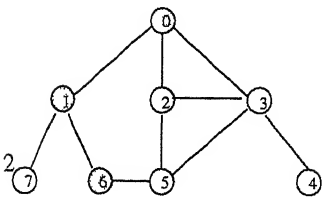
(c) Iteration 2 O={1,2,3,7,6}



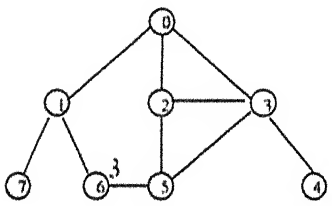
(d) Iteration 3 O={1,2,3,7,6,5}



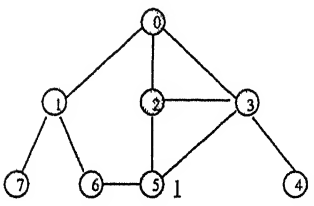
(e) Iteration 4 O={1,2,3,7,6,5,4}



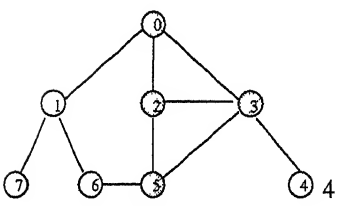
(f) Iteration 5 O={1,2,3,7,6,5,4}



(g) Iteration 6 O={1,2,3,7,6,5,4}



(h) Iteration 7 O={1,2,3,7,6,5,4}



(i) Iteration 8 O={1,2,3,7,6,5,4}

1. All neighbours of root node 0 are assigned 0 as their parent; 0 is added to T.
2. Of possible choices 1, 2 and 3, 1 is arbitrarily chosen, assigned colour 1 and added to T. Nodes 6 and 7 are assigned 1 as their parent.
3. From 2 and 3, node 2 is chosen. **c2b** prevents colour 1 to be assigned to it. So it is assigned colour 2 and added to T. Node 5 is assigned 2 as its parent.
4. Node 3 is chosen, assigned colour 3 and added into T as **c2b** precludes colours 1 and 2 to be assigned to it. Node 4 is assigned 3 as its parent.
5. Node 7 from nodes 6 and 7 is arbitrarily chosen. **c2a** precludes colour 1. So it is assigned colour 2. 7 is then added to T.

In a similar manner nodes 6, 5 and 4 are assigned colours 3, 1 and 4 and added to T in subsequent iterations respectively.

Figure 4.9: Illustrating the operations of the Acc_colour procedure on the sample graph

In the Acc_colour procedure all the nodes in the set $V-\{r\}$ are assigned a colour. Now to colour a node u from this set one needs to investigate the colour of its parent and the colours of its parent's neighbours excluding u itself. So, if the maximum degree present in the graph is ρ, a total of at most ρ investigations need to be carried out to colour a node u. The least available nonconflicting colour can again be found in $O(\rho+1)$ computation steps by keeping a flag for each assignable colour which, from Property 8 can be at most $\rho+1$. Since $N-1$ nodes are to be coloured the procedure Acc_colour will have a worst case running time of $O(\rho N)$. In the Counter_assignment procedure calculating the entry $f(v)$ for each coloured node requires to visit all its children which can be at most ρ thus yielding a complexity of $O(\rho)$. Since the process has to be carried out for all nodes in $V-\{r\}$, this procedure will also have a worst case running time of $O(\rho N)$. Thus the overall time complexity is $O(\rho N)$. □

Property 8 *The set constituting the colours present in the primary frame developed by the algorithm can have a maximum cardinality of $\rho+1$ in a graph having a maximum degree of ρ.*

Proof: For any node v to be coloured the colouring rules **c2a-c2b** require to preclude the colours of its parent and the colours assigned to its parent's neighbours. So the total number of colours precluded can be at most ρ assuming in the worst case that all neighbours of v's parent excluding v itself have been coloured. Thus v can be assigned the remaining $(\rho+1)^{\text{th}}$ colour which proves the above property. □

Property 9 *If d is the degree of the root node, then the length of the accumulation frame used for accumulating the network information at the root node is upper bounded by $\lceil\frac{N-1}{d}\rceil(\rho+1)$ time slots.*

Proof: As has been explained earlier, in each frame the root node gets a new message from each of its neighbours as a consequence of the **Relay rule**. These can be at most d in number. Since $N-1$ new messages need to be accumulated at the root node, the accumulation process can be completed in at most $\lceil\frac{N-1}{d}\rceil(\rho+1)$ time slots, since the primary frame length is upper bounded by $\rho+1$.

4.5.3 Dissemination Algorithm

Having gathered all of the network information at the root node, the question arises as to how to make it available to all other nodes in the network. The algorithm described below does the required dissemination by assigning time slots to the various nodes in the network by colouring the graph $G = (V, E)$ in such a manner that each node can receive the information correctly from its parent in the tree and no node interferes with the transmissions targeted for reception at its children. For this task the algorithm uses the same spanning tree T constructed in the Acc_colour procedure. The algorithm also uses the parent child relationship developed in the Acc_colour procedure.

Starting from the root r the algorithm begins by assigning to it the first colour from the set $\{1, 2, 3, \ldots, \}$ i.e., colour 1. Subsequently, the algorithm proceeds by choosing a particular node out of the uncoloured ones available and colouring it according to the colouring rules **c3** described later. Like in the Acc_colour procedure, in each colouring iteration, we choose an uncoloured node which is the first among the uncoloured ones to have been assigned a parent until this colouring iteration. However in contrast to Acc_colour procedure, in this algorithm nodes having no children are not assigned any colour at all since they would have nothing useful to relay to other nodes in the network. All the messages they will have would already be present at their parent. The algorithm will stop when all nodes having at least a single children in the tree T would have been coloured. For future references we will call Dissem_colour as the procedure performing the colouring part of the algorithm.

Following this algorithm will again give rise to a primary frame consisting of the colours assigned to the nodes by the algorithm. This primary frame if iterated over time will yield the required dissemination frame that can complete the task of gossiping. However in contrast to the centralized spanning tree algorithm in which a node need not transmit a message if it has transmitted it once, in this algorithm it is required that irrespective of whether a node has transmitted a message in the accumulation phase or not it need to transmit it again in the dissemination phase. The reason is that if a node has transmitted a message in the accumulation phase then although it is guaranteed that it will definitely reach its parent, it is not guaranteed that its children will also receive the transmission successfully.

Procedure Dissem_colour(G)

/*

The procedure takes an arbitrary connected undirected graph $G = (V, E)$ as its input. The output is an assignment of colours $c(v)$ to nodes $v \in V$.

*/

```
begin
    for (all v ∈ V) set c(v) ← 0. endfor
    set c(r) ← 1
    while (there is an uncoloured vertex having at least a single child in T) do
        let v be the first node among the uncoloured ones to be assigned a parent
        if (v has at least a single child in G) do
            colour v with the lowest natural number i, such that
                c3a: i has not been assigned to v's parent
                c3b: i has not been assigned to a parent of v's neighbour
        endif
    endwhile
end
```

Figure 4.10: Pseudocode for Dissem_colour procedure

Following are the colouring rules to be followed while assigning colours to the nodes in the graph. In each iteration, the least natural number, c satisfying the following criterion should be assigned to the chosen node v

c3a c is not the colour of v's parent.

c3b c is not the colour of any of v's neighbour parent.

The colouring rules ensure that each node will receive a transmission successfully from its parent in the scheduled slot. The gossip frame will be a concatenation of the accumulation and the dissemination frame.

Now let us apply the Dissem_colour procedure to the sample network shown in Fig-

ure 4.11(a). Further iterations involved are shown in Figures 4.11(b)-(e). In each figure the shaded nodes represent the conflicting nodes for the node chosen in the iteration.

So, primary frame in this case consists of three colours 1, 2 and 3 which can be repeated in time to complete gossiping.

4.5.4 Properties of the Algorithm

Property 10 *The execution time complexity of procedure* Dissem_colour *is* $O(\rho N)$.

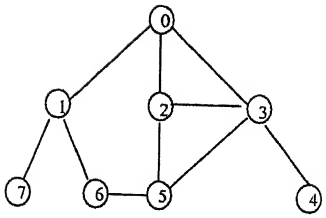
Proof: With simple implementation techniques colour assignment to a node v can be done in at most $O(\rho + 1)$ time steps (ρ denoting the maximum degree present in the graph $G = (V, E)$), since it is required to visit v's parent and parents of v's neighbours which can be at most ρ. By keeping a flag for each colour that can be assigned, the minimum available non conflicting colour can again be found in $\rho + 2$ steps since from Property 11 at most these many colours can be used if coloured using colouring rules **c3**. Therefore, if the cardinality of set V is N the overall worst case time complexity can be $O(\rho N)$. □

Property 11 *The total number of distinct colours assigned by the* Dissem_colour *algorithm is upper bounded by* $\rho + 2$.

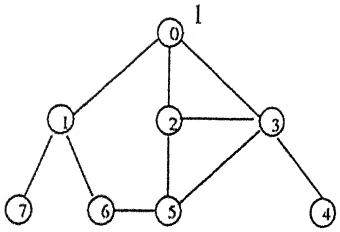
Proof: For any node v to be coloured let us calculate the maximum number of colours that can be conflicting. Since the maximum degree of the network is ρ, node v can have at most ρ neighbours. Since each node can have a single parent in the graph at most $\rho + 1$ colours are precluded by the colouring rules **c3a** and **c3b**. Thus excluding these $\rho + 1$ colours, the remaining $(\rho + 2)^{\text{th}}$ colour can be assigned to v. Since this holds for any $v \in V$ the primary frame length can be no greater than $\rho + 2$. □

Property 12 *If r be the maximum distance from root node to any other node in the network, then the dissemination frame length is upper bounded by* $(N + r - 1)(\rho + 2)$.

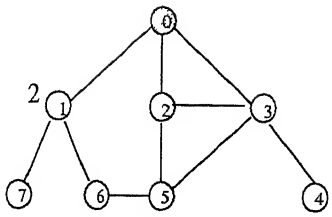
Proof: To complete gossiping, N messages need to be disseminated from the root node to all other nodes in the network. In other words N broadcasts have to be completed in the network to accomplish gossiping. Since, at the beginning of every frame, a new



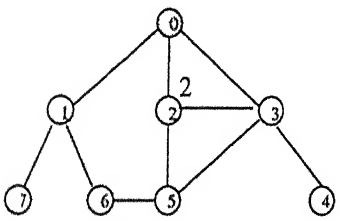
(a) Initially



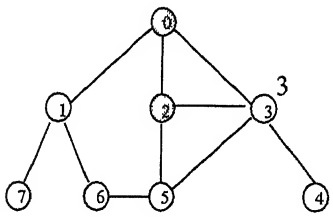
(b) Iteration 1



(c) Iteration 2



(d) Iteration 3



(e) Iteration 4

1. Node 0, the root node is assigned colour 1.
2. Of the three possible choices i.e., nodes 1, 2 and 3 arbitrarily choose node 1, assign colour 2 as **c3a** precludes colour 1.
3. Of nodes 2 and 3, arbitrarily choose node 2, assign colour 2, as **c3a** precludes colour 1.
4. Choose node 3, assign colour 3 because **c3a** precludes colour 1 and **c3b** precludes colour 2.

Since all nodes having at least a single child in the graph have been coloured the procedure ends.

Figure 4.11: Illustrating the operations of the Dissem_colour procedure on the sample graph

(N, R)	10	20	50	100	200
0.3	46	105	351	876	2317
0.4	35	93	267	647	1640
0.5	31	87	250	622	1389
0.6	25	70	231	535	1086
0.7	25	52	123	253	653

Table 4.3: Simulation results depicting the inpractice performance of the "Gather-Scatter Algorithm"

broadcast is initiated by the root node, after a total of $N-1$ frames the last broadcast i.e., the Nth broadcast starts moving from the root node. To have a message from the root node reach all the other nodes requires, in the worst case, a total of r frames. Since the broadcasts move in a pipelined fashion in the network, N broadcasts can be completed in at most $N+r-1$ frames requiring in the worst case a total of $(N+r-1)(\rho+2)$ slots. □

4.5.5 Performance Results

We now present the results of simulation experiments using this algorithm. Table 4.3 shows the number of slots required to complete the gossiping for various values of N and R. Each entry gives the average number obtained by running the algorithm over fifty randomly generated graphs.

As can be seen the algorithm "Gather-Scatter" performs quite well as compared to the "Collision free gossip algorithm" and "Centralized spanning tree" algorithm. Moreover the results show that as R approach towards 1, the number of slots to gossip approaches towards N. This ideally the case should be since for $R = 1$, the graph representing the radio network becomes a complete graph, for which gossiping can be completed in at least N time slots (proved later).

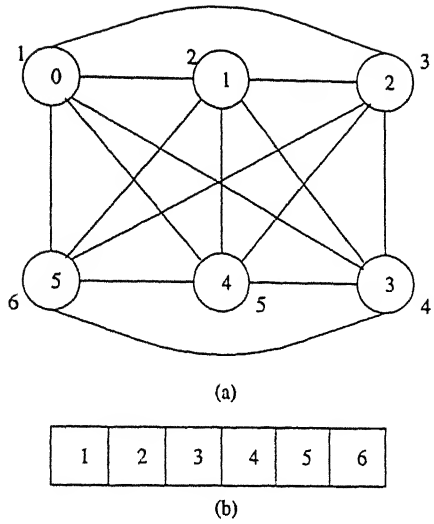


Figure 4.12: Complete Graph (a) Colour assignment (b) Gossip frame

4.6 Gossip Algorithms for Restricted Topologies

In this section we propose gossip algorithms for some restricted radio network topologies. Specifically, we give the schedule for gossiping and the bounds on the gossip frame length for the complete graph, the star network, ring network and a bus network.

4.6.1 Complete Graph

In a complete graph since every node is connected to every other node, a node's transmission is definitely going to reach all other network nodes. However to ensure a successful reception by all the nodes, not more than one node should ever be allowed to transmit in a slot. Violation of this rule would result in a complete loss of information in the network. More specifically, if the rule is not followed then in any scheduling policy no node in the network would ever be able to receive a new message.

Thus, to complete gossiping in a complete graph of N nodes, we would require at least N slots, each slot dedicated to a separate node in the network. In the graph colouring paradigm, this means that each node is to be coloured with a unique colour which is not to be assigned to any other node in the network. This would give rise to a primary frame consisting of N time slots which is sufficient alone to complete the task of gossiping. The colouring mechanism and the required gossip frame for a complete graph of 6 nodes is

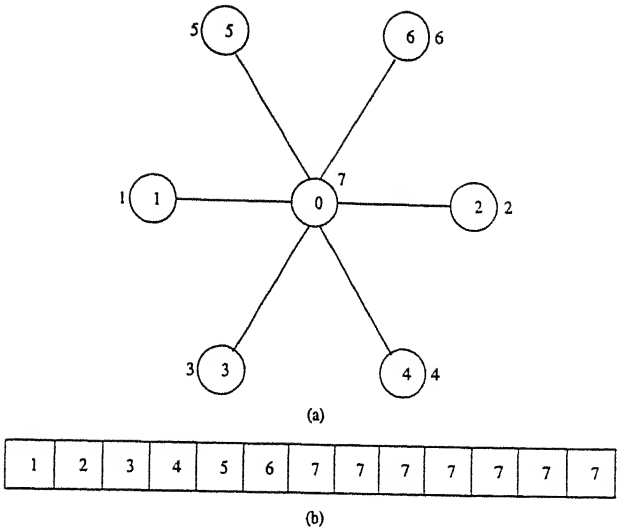


Figure 4.13: Star Network of 7 nodes (a) Colour assignment (b) Gossip frame

shown in Figure 4.12.

4.6.2 Star Network

A star network consists of a central node 0 providing connectivity among the other $N-1$ nodes of the network. No direct path between any two nodes $v_1, v_2 \in V - \{0\}$ exists whereas a path exists between the central node 0 and every other node in $V - \{0\}$. Thus any communication between any two nodes of $V - \{0\}$ would always have to go through the central node.

Therefore, an obvious algorithm to gossip in this case would to be to firstly gather the messages of all the nodes in the central node and then to disseminate them from this central node. Again, as is obvious from the network topology, all the nodes except the central node should be active in the gathering phase. Moreover, in this phase no two nodes can be allowed to transmit in the same slot as it would result in a collision at the central node. In the graph colouring paradigm, this means that we colour each node $v \in V - \{0\}$, with a unique colour c_v not to be assigned to any other node in the network. The entire accumulation process would then take $N - 1$ time slots. Once the accumulation phase is completed we can begin to disseminate and this will obviously take N slots to complete. This is because one transmission by the central node should

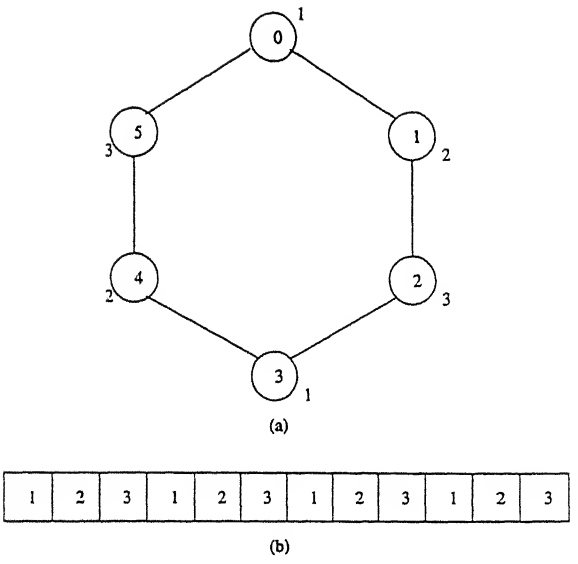


Figure 4.14: Ring Network of 6 nodes (a) Colour assignment (b) Gossip frame

be sufficient to have all the nodes in the network receiving it successfully, provided no other node is allowed to be active during this period. However since, in a slot only a single message can be sent, the entire dissemination phase would require N time slots since there are N messages to be disseminated. In the colouring paradigm this means that we colour the central node with a unique colour c_0 which has not been assigned so far. Repeating it then N times would be sufficient to complete the task of gossiping. So, gossiping in an N-node star network can be completed in exactly $2N-1$ time slots. Figure 4.13 illustrates the situation for a 7-node star network.

4.6.3 Ring Networks

A ring network consists of nodes connected in a cycle with each node in the network connected to two other nodes with which it can communicate. Let us assume that the network consists of N nodes with nodes numbered sequentially from 0 to $N-1$. Again, let us assume that each node can be identified by its sequence number. We propose two algorithms for gossiping in ring networks namely the Colour_ring algorithm and Ring_3m_plus_2 algorithm. The former one can be used to gossip in any N node ring topology, whereas the latter one is a specialized algorithm constructed to gossip in a ring network having $N = 3m+2$ nodes, $m = 1, 2, \ldots$

The algorithm Colour_ring does a distance-2 colouring of the graph G representing the ring network assigning to each node v in the network a colour $c(v)$. It can be easily seen that $3 + k$ colours will be required for a ring having $N = 3m + k$ vertices, where m is an integer and $k = 0, 1$ or 2 The collection of the assigned colours gives rise to a primary frame L (having its cardinality $l = 3 + k$ for $N = 3m + k$), which can be iterated in time to complete gossiping. In the gossip schedule, if the nodes are directed to first transmit their message in the assigned slot and then in subsequent slots transmit the message that have come from the rightmost (leftmost) vertices then it can be easily proved that gossiping can be completed in exactly $l(N-2)$ slots. This is due to the fact that to complete gossiping each node needs to receive a total of $N-1$ messages. So, if the transmissions are scheduled as explained above than each node will receive two new messages in the first frame and then receive one in each subsequent frame, requiring a total of $(N-2)$ frames or $l(N-2)$ slots to complete the task of gossiping. Since each vertex is visited only once in the colouring process the algorithm has a time complexity equal to $O(N)$.

The colouring mechanism and the gossip frame for a ring network of 6 nodes is shown in Figure 4.14

For $N = 3m + 2$ we present a schedule that can complete gossip using fewer slots than the simple relay technique described above for a general ring network. We call this Ring_3m_plus_2 scheduling algorithm.

The Ring_3m_plus_2 algorithm also achieves gossiping by constructing a primary frame which according to certain rules can then be repeated to complete gossiping. The primary frame is obtained by assigning to each node v in the network a colour $c(v)$. The algorithm also requires a root node to be chosen starting from which the colours are assigned to the nodes in the network. Since the ring has circular symmetry, any node in the network can be chosen as the root node. The choice of the root node, say 0 divides the ring into two halves the – the right half containing nodes 1 to $\lfloor \frac{N}{2} \rfloor$ and the left half containing nodes $\lfloor \frac{N}{2} \rfloor + 1$ to $N-1$. The right half will have $\lfloor \frac{N}{2} \rfloor$ nodes and the left half will have $\lceil \frac{N}{2} - 1 \rceil$ nodes.

The basic idea of the algorithm is to have messages from a half propagate towards the root node while making them available to other nodes in the half. When message reaches

Procedure Colour_Ring(G)

/*

The algorithm takes a ring network modelled by the graph $G = (V, E)$ as its input. The output is an assignment of colours $c : V \to \{1, 2, \ldots, \}$.

*/

```
begin
    select any vertex u in G and set x ← u
    if rem(N, 3) = 1, then
        set c(x) ← 4
        x ← vertex to the right of x
    endif
    if rem(N, 3) = 2, then
        set c(x) ← 4
        x ← vertex to the right of x
        set c(x) ← 5
        let x ← vertex to the right of x
    endif
    set c(x) ← 1; set count ← 1
    x ← vertex to the right of x
    while x is not equal to u, do
        c(x) ← count + 1
        x ← vertex to the right of x
        count ← rem(count + 1, 3)
    endwhile
end
```

Figure 4.15: Pseudocode for Colour_ring procedure

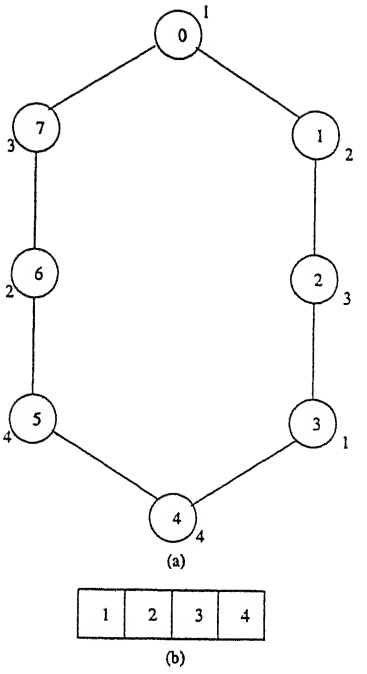


Figure 4.16: Ring Network of 8 nodes (a) Colour assignment (b) Primary frame

the root node. the root node then makes it available to the nodes in the other half. This can be achieved as follows. Consider nodes in the right half The colouring needs to be done in such a manner that transmission by a node v can be received successfully by nodes $v+1$ and $v-1$, in the half. The transmission scheduling should be such that each node first transmits its own message and then transmits the message of any of the nodes from $v+1$ to $\lfloor \frac{N}{2} \rfloor$. (Note that node $\lfloor \frac{N}{2} \rfloor$ does not have to relay the transmission from node $\lfloor \frac{N}{2} \rfloor + 1$.) If such a message does not exist, any other message not transmitted earlier can be chosen and transmitted. Similar arguments can be made for the nodes in the left half. It can also be seen that the last nodes in each half, i.e., nodes $\lfloor \frac{N}{2} \rfloor$ and $\lfloor \frac{N}{2} \rfloor + 1$ need to transmit only once and hence if the colour assigned to them is not assigned to any other node in the network this colour needs to be repeated only once in the gossip frame. The transmission schedule for the root node is that it can transmit any message that it has not transmitted earlier in the gossip frame.

This colouring mechanism would require exactly four colours to colour the graph of the ring network, with one colour i.e., 4 assigned to the last nodes in each half. This colour is to be repeated only once in the gossip frame.

Procedure Ring_3m_plus_2(G)

/*

The algorithm takes a ring network modelled by an undirected graph $G = (V, E)$ as its inpu

The output is an assignment of colours $c : V \rightarrow \{1, 2, \ldots, \}$

*/

begin

select any vertex 0 in the ring as root.

set $c(0) \leftarrow 1$

set $colourvar \leftarrow 1$; set $rnodecount \leftarrow 0$

set $x \leftarrow 1$

while $(rnodecount < \lfloor \frac{N}{2} \rfloor - 1)$

$c(x) \leftarrow colourvar + 1$

$colourvar \leftarrow \text{rem}(colourvar + 1, 3)$

$x \leftarrow x + 1$

$rnodecount \leftarrow rnodecount + 1$

endwhile

set $colourvar \leftarrow 0$; set $lnodecount \leftarrow 0$

set $x \leftarrow N - 1$

while $(lnodecount < \lceil \frac{N}{2} - 1 \rceil - 1)$

$c(x) \leftarrow (3 - colourvar)$

$colourvar \leftarrow \text{rem}(colourvar + 1, 3)$

$x \leftarrow x - 1$

$lnodecount \leftarrow lnodecount + 1$

endwhile

$c(\lfloor \frac{N}{2} \rfloor) \leftarrow 4$; $c(\lfloor \frac{N}{2} \rfloor + 1) \leftarrow 4$;

end

Figure 4.17: Pseudocode for Ring_3m_plus_2 procedure

Since each vertex is considered only once in the colouring process the algorithm has a time complexity equal to $O(N)$. Let us now calculate the length of the gossip frame resulting from the use of this algorithm. The colouring mechanism and the rule for transmission ensures that the root gets at least one new message in every frame. Thus a new broadcast can be initiated from the root node in every frame and so N broadcasts can be initiated in N frames. The messages propagate along both halves in a pipelined fashion with the primary frame length corresponding to the time of a pipeline step. Again, in this scheduling algorithm, if N is even, a broadcast initiated by the root takes $N-3$ time slots to reach all the nodes in the network when the nodes have nothing else to transmit. (It takes $N-2$ slots if N is odd). Thus, under this algorithm it would take exactly $3(N-1)+(N-2)+1 = 4N-4$ slots to gossip when N is odd and $3(N-1)+(N-3)+1 = 4N-5$ slots when N is even.

For a ring of 8 nodes the colouring scheme and the required primary frame is as shown in Figure 4.16.

4.6.4 Bus Networks

A bus network consists of nodes aligned up in a line as shown in Figure 4.18(a). To describe our gossip algorithm in this case, let us assume that the network consists of N nodes with nodes sequentially numbered 0 to $N-1$ from the left. Our gossip algorithm requires that a node in the network be chosen as the root node. In general, any node in the network could be chosen as the root and the algorithm would still be correct. However, the time required to gossip would depend on this choice. More specifically, choosing the central node as the root node would yield the shortest time to complete gossip using our scheduling algorithm. In the following, we will assume that the root node will be node $\lfloor \frac{N}{2} \rfloor$ for odd N and node $(\frac{N}{2}-1)$ for even N. The root node divides the bus into two halves with the left half consisting of nodes with sequence numbers lesser than the root node and the right half consisting of nodes with sequence numbers greater than the root node. Thus the line can be viewed as shown in Figure 4.18(b). Let us now explain in detail the approach used by us to gossip in a bus network. For ease of presentation, in the right half, we say, that nodes above node v in this half will be those with sequence numbers between r and v and the nodes below node v are those numbered

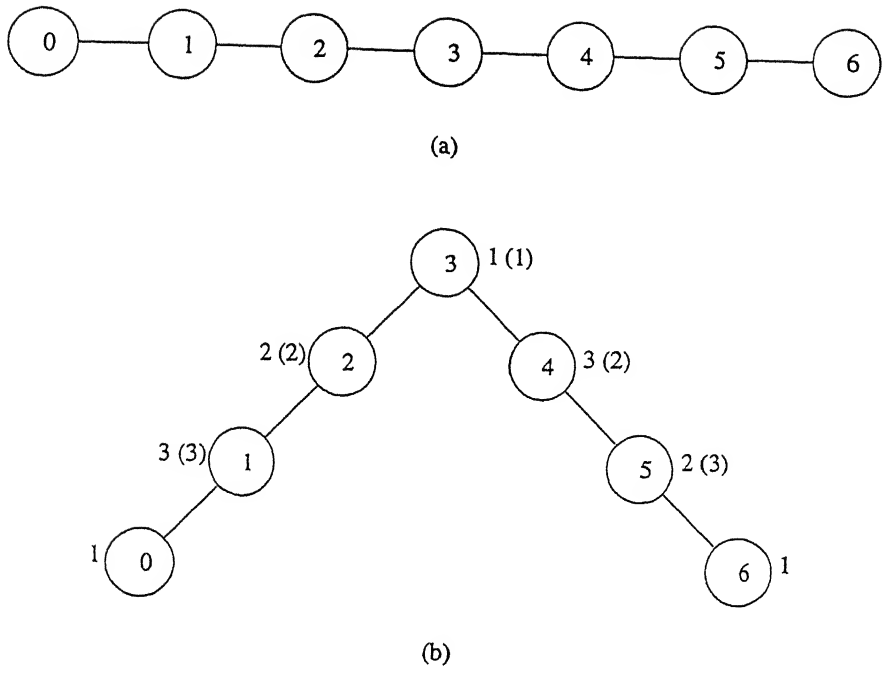


Figure 4.18: (a) Bus network of 7 nodes (b) Colour assignment: $c_1(c_2)$

$v+1$ to $N-1$. Similarly, if v were in the left half, nodes 0 to $v-1$ would be considered to be below it and nodes $v+1$ to r would be considered to be above it. These relations are illustrated in Figure 4.18(b)

The basic idea in this algorithm is to have messages from both halves of the bus relayed along the bus till they reach the root node and while reaching the root node, the messages will also be made available to the nodes of that half according to the relay rule described later. The root node then relays the messages of each half to the nodes in the other half. This is achieved by dividing the task into phases. In the first phase nodes are scheduled to transmit in such a manner that the messages from each half of the bus move towards the root node as quickly as possible. This is achieved by assigning to each node v in the network, $0 \leq v \leq N-1$, a colour $c_1(v)$ according to which its transmissions should be scheduled. The colours are assigned in the sequence $1, 2, 3, 1, 2, 3, \cdots$ from the root node downwards on the left half and the sequence $1, 3, 2, 1, 3, 2, \cdots$ from the root node downwards on the right half. This is shown in Figure 4.18(b). This colouring scheme ensures that each node's transmission can reach the node above it successfully. In this phase the relay rule for the nodes is to transmit their own messages in the first chance and in subsequent chances transmit the messages of the nodes below it. If no such message

is available then the node can transmit any message that it has not transmitted before. This relay rule ensures that in this phase, the messages from each half move towards the root node as quickly as possible. The primary frame consisting of the three colours 1, 2 and 3 is repeated $\lfloor \frac{N}{2} \rfloor$ times to complete an accumulation at the root node. This is because node $r-1$ will have $\lceil \frac{N}{2} - 2 \rceil$ nodes below it and node $r+1$ will have $\lfloor \frac{N}{2} - 1 \rfloor$ nodes below it and these nodes will require to transmit their messages and then relay the messages of the nodes below them.

At the end of the first phase all the messages would have finished their journey upwards towards the root node. To complete the downward journey into the other half of the bus, the transmissions need a second phase. In this phase the transmissions must be scheduled such that the messages are relayed downwards from each node. In this phase each node v (excluding nodes 0 and $N-1$) is assigned a colour $c_2(v)$ in such a way as to avoid collisions at nodes below it. This is achieved by assigning the colours in the sequence $1, 2, 3, 1, 2, 3, \cdots$ from the root node downwards on both halves of the bus. This is shown in Figure 4.18(b). For $N \leq 5$ the number of colours used may be less than 3. We will denote by F_2, the number of colours used in the primary frame in this phase. In this phase the relay rule for the nodes is that they can transmit any message they have not transmitted before in either the first or the second phases i.e., a node need to transmit a message only once before gossip is achieved. The gossip frame will consist of frames from the first and the second phases. Since each vertex is visited only once in the colouring process the algorithm has a time complexity equal to $O(N)$.

A total of $(\lfloor \frac{N}{2} \rfloor)$ broadcasts would have been initiated by the root node in the first phase since it had that many chances to transmit in the first phase. Also, the root node, after completion of the first phase would have all the N messages. So the root node needs to transmit at least $(N - \lfloor \frac{N}{2} \rfloor)$ times in the second phase requiring the primary frame to be repeated at least $(N - \lfloor \frac{N}{2} \rfloor - 1)$ times before the last message begins to move downwards from the root node. For this last message to reach node the end node a minimum of $\lfloor \frac{N}{2} \rfloor$ time slots would be required. Therefore, a total of $3(\lfloor \frac{N}{2} \rfloor) + F_2(N - \lfloor \frac{N}{2} \rfloor - 1) + (\lfloor \frac{N}{2} \rfloor)$ slots would be needed to complete gossiping in a bus network of N nodes.

Procedure Bus_colour(G)

/*

The procedure takes a bus network modelled by an undirected graph $G = (V, E)$ as its input

The output is an assignment of colours $c_1, c_2 : V \rightarrow \{1, 2, \ldots, \}$.

*/

begin

set $r \leftarrow \lfloor \frac{N}{2} \rfloor$ for odd values of N and $r \leftarrow \frac{N}{2} - 1$ for even values of N.

set $count1 \leftarrow 0$; set $x \leftarrow r$

while x is greater than or equal to 0, do

$c_1(x) \leftarrow count1 + 1$

if x is not equal to 0 then, $c_2(x) \leftarrow count1 + 1$ endif

$count1 \leftarrow \text{rem}(count1 + 1, 3)$

$x \leftarrow x - 1$

endwhile

set $count1 \leftarrow 0$; set $x \leftarrow r + 1$

while x is less than or equal to $N - 1$, do

$c_1(x) \leftarrow (3 - count1)$

if x is not equal to $N - 1$ then, $c_2(x) \leftarrow \text{rem}(count1 + 1, 3) + 1$ endif

$count1 \leftarrow \text{rem}(count1 + 1, 3)$

$x \leftarrow x + 1$

endwhile

end

Figure 4.19: Pseudo code for procedure Bus_colour(G)

Chapter 5

Conclusions and Future Work

5.1 Summary and Conclusions

In this thesis we have proposed some algorithms to gossip in arbitrarily connected multihop radio networks. The algorithms can be used to exchange control information such as connectivity and routing information among the network stations irrespective of the topology the radio network might have. In addition, the inpractice performance of all the proposed algorithms is provided by running the algorithms on a large number of randomly generated graphs representing multihop radio networks. For a realistic simulation, we have used a technique that is widely used to study radio networks. In this simulation model, the transmitters of all the nodes are assumed to have a fixed range R which means that all nodes in this range will be capable of receiving the transmitted signal. We have studied the algorithms for various transmission ranges and number of nodes in the network. The values of R chosen are those that are commonly found in most existing packet radio networks.

The first algorithm in this context is the Collision Free Gossip algorithm which is a multipurpose algorithm in the sense that the primary frame developed by this algorithm can be used for carrying out other network tasks like point to point communication among nodes. This is due to the fact that the primary frame in this algorithm is constructed in a manner that prevents all kinds of collisions in the network. However, the price to be paid is in terms of the number of slots that the algorithm takes to complete gossiping. The Collision Free Gossip algorithm does not significantly improve the size of the primary

frame in subsequent iterations in the gossip frame. As can be seen from Table 4.1, for a given value of N as R increases, the number of slots taken by the Collision Free Gossip algorithm to gossip also increases. This is in contrast to the Centralised Spanning Tree and the Gather-Scatter algorithms in which the number of slots to complete gossip decreases as it should ideally be. This is because as R increases, the graph corresponding to the network approaches a complete graph. This can be understood by the fact that an increase in the value of R results in a dense network which in turn requires a large primary frame. In the Collision Free Gossip algorithm, this primary frame does not undergo significant improvements in subsequent iterations resulting in a large gossip frame. However, in the other two algorithms, not only is the initial primary frame length smaller, but also the improvements that the primary frame undergoes after the first iteration is significant and contributes in greatly reducing the length of the gossip frame. In both the Centralised Spanning Tree and the Gather-Scatter algorithms, a tree is constructed and as the graph becomes dense, the depth of this tree decreases. This allows the messages to reach the end nodes quicker. Moreover, the leaf nodes in the tree are also shielded in these algorithms from transmitting but once. This is in contrast to the Collision Free Gossip algorithm in which the messages do not propagate along trees.

Among the Centralised Spanning Tree and the Gather-Scatter algorithms, we observe that the latter performs better. However, an obvious disadvantage of this algorithm is that the root node is responsible for scattering all of the messages and can become congested.

As regards the relay rule that the algorithm may adopt for selecting a feasible message out of the several ones possible in its buffer, it can be seen from Tables 4.1 and 4.2 that random order of service (ROS) exhibits the best overall performance with FCFS, IOS and LCFS coming next in order. The observation that LCFS and IOS perform poorer than FCFS and ROS, is consistent with the well known fact that queues in which such a choice of arbitrarily selecting a feasible message exists, a relatively better performance is obtained from a selection method that tends to avoid having some message consistently blocked by others. The observation that ROS is better than FCFS is consistent with a similar observation made by Topkis [54] where he considered gossiping in wireline networks. It can also be seen that a node giving priority to its own message and then

follow the relay rules results in a better performance as compared to the procedure of strictly following the rules.

In addition to providing gossip algorithms for arbitrarily connected multihop radio networks, we have also proposed algorithms for gossiping in some regular topology networks like star, ring and bus networks.

To summarize, the main contributions of this thesis include:

- A reasonably exhaustive survey of existing gossip algorithms in various information exchange models.
- Description and analysis of several approximation algorithms to gossip in radio networks for both arbitrary and restricted topologies.
- Formulation of a realistic experimental model and providing the inpractice performance of various heuristics based on this model.

5.2 Future Work

We are the first to study realistic gossiping in multihop radio networks. For arbitrary graphs, it is known that optimal scheduling algorithms are difficult to find. Hence, only approximate algorithms can be given. Although, the Gather-Scatter algorithm has an excellent inpractice performance, there is possibly room to improve upon these algorithms. Also, the optimal gossip schedules for regular topology networks may be found by further improving our algorithms.

The algorithms described in this thesis are centralised algorithms in the sense that the algorithm obtains the entire graph as its input. To respond acceptably to changes in the topology and for reduced vulnerability, it is desirable that the algorithm be distributed. In a distributed algorithm, the stations participate in the computation of the schedule by exchanging messages. An important future direction would be to consider efficient distributed implementations of the proposed algorithms.

Bibliography

[1] N. Abramson, "Packet switching with satellites," *Proceedings of the National Computing Conference,* pp. 695-702, June 1973.

[2] E. Arikan, "Some complexity results about packet radio networks," *IEEE Transactions on Information Theory,* Vol. IT-30, pp. 910-918, July 1984.

[3] A. Bagchi, S. L. Hakimi, L. Mitchem and E. Scmeichel, "Parallel algorithms for gossiping by mail," *Information Processing Letters* Vol. 34, pp. 197-202, 1990.

[4] A. Bagchi, E. F. Schmeichel and S. L. Hakimi, "Parallel information dissemination by packets," *SIAM Journal of Computing,* Vol 23, pp. 355-372, 1994.

[5] A. Bavelas, "Communication patterns in task-oriented groups," *J. Acoust. Soc. Amer.,* Vol 22, pp. 725-730, 1950.

[6] J. C. Bermond, L. Gargano, A. Rescigno and U. Vaccaro "Fast Gossiping by short messages," *Lecture Notes in Computer Science, Springer-Verlag,* Vol. 944, pp. 135-146, 1995.

[7] J. C. Bermond, T. Kodate and S. Perennes. "Gossiping in Cayley graphs by packets," *Lecture Notes in Computer Science, Springer-Verlag,* Vol. 1120, pp. 301-315, 1995.

[8] D. Bertsekas and R. Gallager, "Data Networks," *Englewood Cliffs, NJ: Prentice-Hall,* 1992

[9] B. Bollobas, "Random Graphs," *Academic Press Inc., London,* 1985.

[10] I. Chlamtac and S. Kutten, "A spatial reuse TDMA/FDMA for mobile multihop radio networks," *Proceedings of the INFOCOM,* March 1985.

[11] I. Chlamtac and S. Lerner, "A link allocation protocol for mobile multihop radio networks," *Proceedings of the GLOBECOM,* Dec. 1985.

[12] I. Chlamtac and S. Lerner, "Fair algorithms for maximal link activation in multihop radio networks," *IEEE Transactions on communications,* Vol. COM-35, pp. 739-746, July 1987.

[13] A. M. Chou and V. O. K. Li, "Fair spatial TDMA channel access protocols for multihop radio networks," *Proceedings of the INFOCOM,* 1989.

[14] I. Cidon and M. Sidi, "Distributed assignment algorithms for multihop radio networks," *IEEE Transactions on Computers,* Vol.38, no. 10, pp. 1353-1361, Oct. 1989.

[15] G. Cybenko, D. W. Krumme and K. N. Venkataraman, "Gossiping in minimal time," *SIAM Journal of Computing,* Vol. 21, pp. 111-139, 1992.

[16] G. Cybenko, D. W. Krumme and K. N. Venkataraman, "Simultaneous broadcasting in multiprocessor networks," *Proc. International Conference on Parallel Processing,* pp. 555-558, 1986.

[17] A. Ephremedis and T. V. Truong, "Scheduling Broadcasts in multihop radio networks," *IEEE Transcations on communications,* Vol. COM-38, pp. 456-460, Apr. 1990.

[18] S. Even, O. Goldreich, S. Moran and P. Tong, "On the NP-completeness of certain network testing problems," *Networks,* Vol. 14, pp. 1-24, 1984.

[19] S. Even and B. Monien, "On the number of rounds necessary to disseminate information," *Proccedings of the Ist ACM Symposium on Parallel Algorithms and Architectures,* pp. 318-327, June 1989.

[20] S. Even, "Graph Algorithms," *Computer Science Press,* 1979.

[21] R. C. Entringer and P. J. Slater, "Gossips and telegraphs," *Journal Franklin Institute,* Vol. 307, pp. 353-360, 1979.

[22] A. M. Farley, " Minimum time line broadcast networks," *Networks,* Vol. 10, pp. 59-70, 1980.

[23] A. M. Farley and A. Proskurowski, "Gossiping in grid graphs," *J. Combin. Inform. Systems Sci.,* Vol. 5, pp. 161-172, 1980.

[24] R. Feldmann, J. Hromkovic, S. Madhavapeddy, B. Monien and P. Mysliwietz, "Optimal algorithms for dissemination of information in generalized communication modes," *Lecture Notes in Computer Science,* Vol. 605, pp. 115-130, Springer Verlag 1992.

[25] P. Fragopoulou, S. G. Akl and H. Meijer, "Optimal communication primitives on the generalized hypercube network," *Journal of Parallel and Distributed Computing.* Vol. 32, no. 2, pp. 173-187, 1996.

[26] P. Fragopoulou and S. G. Akl, "A framework for optimal communication on a subclass of the Cayley Class based networks," *Proc. 14th Annual IEEE conference on Computers and Communications,* pp. 241-248, 1995.

[27] P. Fraigniaud and E. Lazard, "Methods and problems of communication in usual networks," *Discrete Applied Mathematics,* Vol. 53, pp. 79-134, 1994.

[28] H. Frank, I. Gitman, and R. Van Slyke, "Packet radio system-network considerations," *Proceedings of the National Computing Conference,* pp. 217-231, May 1975.

[29] W. M. Gentleman, "Some complexity results for matrix computations on parallel processors," *J. Assoc. Comput. Mach.,* Vol. 25, pp. 112-115, 1978.

[30] S. Ginn, "Personal Communications: Expanding the freedom to communicate," *IEEE Communications Magazine,* pp. 30-39, Feb 1991.

[31] W. K. Hale, "Frequency assignment: Theory and applications," *Proceedings of the IEEE,* Vol. 68, no. 12, Dec. 1980.

[32] A. Hajnal, E. C. Milner and E. Szemeredi, "A cure for the telephone disease," *Canad. Math. Bull.,* Vol. 15, pp. 447-450, 1972.

[33] J. Hromkovic, C. D. Jeschke and B. Monien, "Optimal algorithms for dissemination of information in some interconnection networks," *Proc. MFCS 90, Lecture Notes in Computer Science,* Vol. 452, pp. 337-346, Springer Verlag 1990.

[34] R. E. Kahn, "The organisation of computer resources into a packet radio network," *Proceedings of the National Computing Conference,* pp. 177-186, May 1975.

[35] L. Kleinrock and S. S. Lam, "Packet switching in a slotted satellite channel," *Proceedings of the National Computing Conference,* pp. 603-710, June 1973.

[36] W. Knodel, "New gossips and telephones," *Discrete Mathematics,* Vol. 13, pp. 95, 1975

[37] D. W. Krumme, "Fast Gossiping for the hypercube," *SIAM Journal of Computing,* Vol. 21, no. 2, pp. 365-380, 1992.

[38] R. Labahn, "The telephone problem on trees," *Elektron. Informationsverarb. u. Kybernet.,* Vol. 22, pp. 475-485, 1980.

[39] H. G. Landau, " The distribution of completion times for random communication in a task oriented group," *Bull. Math. Biophys.,* Vol. 16, pp. 187-201, 1954.

[40] E. L. Lloyd and S. Ramanathan, "On the complexity of link scheduling in multihop radio networks," *Proceedings of the 26th Conference on Information Sciences and Systems,* pp. 491-495, March 1992.

[41] D. W. Matula, G. Marble and J. F. Issacson, "Graph colouring algorithms," *Graph Theory and Computing,* New York: Academic, 1982.

[42] R. Nelson and L. Kleinrock, "Spatial TDMA: A collision free multihop channel access protocol," *IEEE Transcations on communications,* Vol. COM-33, pp. 934-944, Sep. 1985.

[43] M. Post, P. Sarachik and A. Kershenbaum, "A biased greedy algorithm for multihop radio networks," *Proceedings of the Conferencce on Information Sciences and Systems,* pp. 564-572, 1985.

[44] R. Ramaswami and K. K. Parhi, "Distributed scheduling of broadcasts in a radio network," *Proceedings of the INFOCOM,* 1989.

[45] Reuven Bar Yehuda, "Personal Communication," *Technion Institute of Technology,* Isreal.

[46] K. Ravishankar and S. Singh, "Gossiping on a ring with radios," *Parallel Processing Letters,* Vol. 6, no. 1, pp. 115-126, 1996.

[47] K. Ravishankar and S. Singh, "Broadcasting on [0, L]," *Discrete Applied Mathemetics,* Vol. 54, no. 1, pp. 115-126, 1995.

[48] N. Shacham and J. Westcott, "Future directions in packet radio architectures and protocols," *Proceedings of the IEEE,* Vol. 75, pp. 83-99, 1987.

[49] A. Shimbel, "Applications of the matrix algebra to communication nets," *Bull. Math. Biophys,* Vol. 13, pp. 165-178, 1951.

[50] J. A. Silvester, "Perfect scheduling in multihop broadcast radio networks," *Proceedings of the ICCC,* 1990.

[51] S. Singh and M. A. Sridhar, "Gossiping on interval graphs," *7th International Parallel Processing Symposium,* pp. 338-343, 1993.

[52] F. A. Tobagi, "Multi-access protocols in packet communication networks," *IEEE Transactions on communications,* Vol. COM-28, pp. 468-488, Apr. 1980.

[53] F. A. Tobagi, "Modelling and performance analysis of multihop packet radio networks," *Proceedings of the IEEE,* Vol. 75, no. 1, pp. 135-155, Jan. 1987.

[54] D. M. Topkis, "All-to-all broadcast by flooding in communication networks," *IEEE Transactions on Communications,* Vol. 38, no. 9, pp. 1330-1333, 1989.

[55] B. Walke and Th. Hellmich, "Highly reliable channels for short-range mobile radio networks," *Proceedings of the ICCC,* 1990.

[56] D. B. West, "A class of solutions to the gossip problem part I," *Discrete Mathematics,* Vol. 39, pp. 307-326, 1982.

[57] D. B. West, "A class of solutions to the gossip problem part II," *Discrete Mathematics,* Vol. 40, pp. 87-113, 1982.

[58] D. B. West, "A class of solutions to the gossip problem part III," *Discrete Mathematics,* Vol. 39, pp. 285-310, 1982.